YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SES HABERLEŞME SİSTEMLERİ İÇİN DİNAMİK ERİŞİM KONTROLÜ TASARIMI VE UYGULAMASI

Bilgisayar Mühendisi Burak DOĞRUÖZ

FBE Bilgisayar Mühendisliği Anabilim Dalında Hazırlanan

YÜKSEK LİSANS TEZİ

Tez Danışmanı : Yrd. Doç. Dr. A. Gökhan YAVUZ (Y.T.Ü.)

İSTANBUL, 2007

İÇİNDEKİLER

Sayfa

KISALTMA LİSTESİ

API	Application Programming Interface
CTI	Computer Telephony Integration
FCT	Fixed Cellular Terminal
FIFO	First In First Out
GPL	General Public Licence
GSM	Global System for Mobile communications, Groupe Spécial Mobile
IP	Internet Protocol
ISO	International Standards Organisation
ITU	International Telecommunication Union
JTDS	JAVA Tabular Data Streams
LCR	Least Cost Routing
MSDE	Microsoft Desktop Engine
PBX	Private Branch Exchange
PSTN	Public Switch Telephony Network
SPIT	Spam over Internet Telephony
VoIP	Voice Over Internet Protocol

ŞEKİL LİSTESİ

ÇİZELGE LİSTESİ

ÖNSÖZ

Tüm hayatım boyunca bana her konuda destek olan aileme, başta Mazlum KUTLAR ve Tolga GÜNDÜZ olmak üzere, tez çalışmamda ihtiyaç duyduğum ortam ve imkanları sağlayan çalışma arkadaşlarıma, yüksek lisans eğitimi ve tez geliştirme süresince yardımlarını, bilgisini ve zamanını esirgemeyen danışman hocam Yrd. Doç. Dr. A. Gökhan YAVUZ'a ve bölüm başkanımız Prof. Oya KALIPSIZ'a teşekkürü borç bilirim.

ÖZET

Günümüzün en önemli iletişim araçlarından olan telefon, yalnız gündelik hayatın değil, iş dünyası süreçlerinin de en temel cihazlarından biridir. Özellikle gelişen teknoloji ve iletişim sektöründe yapılan düzenlemeler sonucunda azalan görüşme maliyetleri sayesinde kurumlar, iş ilişkilerinde telefon kullanımına verdikleri ağırlığı daha da arttırmıştır. Ekonomik gelişmeye ek olarak, yakalanan teknolojik gelişmeler sayesinde geleneksel anlamdaki telefon konuşması kavramına yeni fonksiyonlar eklemiş, bu da mevcut iletişim deneyiminin sağladığı verime önemli bir katkı yapmıştır.

Sesli iletişimin iş dünyasındaki artan önemi ve bahsedilen gelişmeler, mevcut iletişim sistemlerinde bir takım yeni ihtiyaçların ortaya çıkmasına neden olmuştur. İş süreçlerinin düzgün işleyebilmesi ve maliyetlerinin düşürülmesi gibi amaçlarla sesli iletişim trafiğinin izlenmesi, analiz edilmesi ve kontrol edilmesi, önemli bir fonksiyon haline gelmiştir.

Sesli haberleşme trafiğinin düzenlenmesi sayesinde yanlış bağlanan telefonlar ve gereğinden uzun süren görüşmelerin neden olduğu iş performansı kayıplarının önüne geçilebileceği gibi, özellikle VoIP (Voice over IP) altyapısını kullanılarak yapılan telefonla pazarlama (telemarketing) ve genellikle SPIT (Spam over Internet Telephony) [1] olarak tanımlanan istenmeyen telefon trafiği gibi iş verimini düşüren faaliyetler de engellenebilmektedir.

Bu tez çalışmasının amacı, yoğunluklu olarak telefonla iş geliştiren kurumlar başta olmak üzere, bir organizasyonun telefon trafiğini kontrol edecek bir sistem gerçeklemek ve bu sistem sayesinde, uygun görülmeyen görüşmelerin engellenmesi veya belirli kurallar dahilinde düzenlenmesi gibi bir takım dinamik fonksiyonların kullanımı ile organizasyondaki iş veriminin arttırılmasını sağlamaktır.

Tasarlanan sistemin gerçeklendiği uygulama, platform ve kullanım alanından bağımsız bir tasarım esas alınarak farklı sesli iletişim sistemleriyle birlikte çalışabilecek şekilde geliştirilmiştir. Bu sayede, çalışmanın ortaya koyduğu fonksiyonların farklı teknik altyapılara sahip sistemlerde birbirinden bağımsız amaçlarla kullanılabilmesine imkan sağlanmakla birlikte mevcut sesle iletişim sistemlerinin barındırdığı kısıtlı özellikleri genişleterek, bu sistemleri kullanan kurumların kendilerine has ihtiyaçlarını karşıyabilecek yeni özelliklerin bu sistemlere kazandırılması mümkün kılınmıştır.

Tez çalışması sonucunda tasarlanan sistem, JAVA [2] programlama dili kullanılarak gerçeklenmiştir. Gerçekleştirilen uygulama ile bir kurumdaki iletişim trafiğine yönelik senaryolar belirlenmiş ve bu senaryolar çerçevesinde, mevcut açık kaynak kodlu bir IP PBX yazılımı olan Asterisk [3] kurulumu üzerinden yapılan görüşmelere yönelik kurallar tanımlanarak iletişim trafiğinin izlenmesi ve düzenlenmesi sağlanmıştır.

Anahtar kelimeler: telefon, sesli iletişim, yönetim, kontrol, verim

ABSTRACT

Being one of today's most important communicaton tools, telephone is not just a basic appliance of daily life, but also business processes. Especially, as a result of lower costs caused by improving technologies and established regulations in communication sector, organizations increased their telephone usage for their business relations. On the side of economical improvement, the technological improvement added new functions and concepts to the traditional telephony and thus contributed to the current communication experience and efficiency.

The increasing importance of voice communication in business world and the improvements dealed above caused some requirements to arise. For performing business processes correctly and lowering costs more, it has become an important function to watch, analyze and control voice communication traffic.

By organizing communication traffic, performance lost caused by calls that proceeds longer than necessary and miscalls will be prevented as well as unwanted telephony traffic over VoIP (Voice Over IP) infrastructure that is generally defined as SPIT (Spam over Internet Telephony) [1] which reduces business efficiency.

The purpose of this study is to realize a system that will control the telephony traffic of organizations, particularly the ones that manage business jobs by telephone, and to increase business efficiency by blocking inappropriate calls or arranging them by certain rules using dynamic functions with the help of this system.

The application which realizes the system defined on this study is developed in a way that is compatible with different voice communication systems and is based on a design that is independent from platform and area of application. Thus, it becomes possible to use the defined functions for independent and varied purposes as well as extending limited features of present voice communication systems and bringing new and unique features that may be required by an organization.

The system designed in this study is realized by an application developed using JAVA [4] programming language which was chosen to provide platform independency. To test the application results, a set of scenarios concerning voice communication traffic on a typical organization were defined and calls being carried on a present open source software PBX Asterisk [3] installation were tracked and regulated by determining rules within these scenarios.

Keywords: telephone, voice communication, management, control, efficiency

1. GİRİŞ

Çağımızın iletişim teknolojilerindeki gelişme, günlük hayatı ve yaşam tarzını da doğrudan etkilemektedir. Özellikle sesli iletişim araçlarındaki gelişme ve gittikçe daha da büyüyen iletişim sektöründeki amansız rekabetin getirdiği düşük maliyetler, başta mobil telefonlar olmak üzere tüm sesli iletişim araçlarının, artan bir eğilimle geniş bir kullanım alanı bulmasını sağlamıştır. Bu geniş kullanım alanı, iş süreçlerinde telefon kullanımına verilen ağırlığı daha da arttırarak mevcut iletişim sistemlerine yönelik yeni ihtiyaçların ortaya çıkmasına neden olmuştur. İş süreçlerinin düzgün işleyebilmesi ve maliyetlerinin düşürülmesi gibi amaçlarla, görüşmelerin analiz ve kontrolü, kurumlar için önemli bir gereksinim haline gelmiştir.

İletişim sistemi geliştiricileri ve üreticileri, yeni nesil ürünlerine ekledikleri fonksiyonlarla ortaya çıkan bu gereksinimleri karşılamaya çalışsa da, ortaya konulan çözümler çoğunlukla kısıtlı, yerel ve markaya bağımlı olmakta, sürekli değişen kurum ihtiyaçlarını karşılayacak dinamikliği içermemektedir. Özellikle farklı marka ve sistemlerin bir arada kullanıldığı kurumlarda en temel özelliklerin bile ortak bir paydada birleştirilmesi mümkün olamamaktadır. Özetle, farklı alt yapı ve platformlardaki iletişim sistemlerinin geneline yönelik ve geniş bir fonksiyon kümesine sahip bir çözüm ortaya konmadığından bu alandaki ihtiyaç tam anlamıyla karşılanamamaktadır.

Bu tez çalışmasında, günümüzün değişen ve gelişen telefon kullanım alışkanlıklarıyla beraber ortaya çıkan, telefonların yanlış bağlanması, iş performansı kayıplarını oluşturacak şekilde görüşmelerin gereğinden uzun sürmesi gibi bir takım sorunların üstesinden gelmek ve ticari işletmelerin gittikçe artan telefonla iş geliştirme faaliyetlerinde, elde ettikleri verimi arttırmak için bir sistem tasarlanmış ve bu tasarımı temel alan platform-bağımsız bir uygulama gerçeklenmiş, belirlenen senaryolar için elde edilecek faydalar değerlendirilmiştir.

Tasarlanan sistemin yapısı, çalışma şekli ve sunduğu fonksiyonlar göz önüne alınarak uygulanabileceği beş temel senaryo belirlenmiştir. Bu senaryolar; kurumlarda görüşme disiplininin sağlanmasına yönelik denetim uygulamaları, ön ödemeli servislerin devreye alınarak mevcut sesli iletişim sistemine katma değerli yeni hizmetlerin eklenmesi, numara taşınabilirliği uygulamaları sonucu ortaya çıkan ek maliyetlerin giderilmesi, numara yönlendirme uygulamalarıyla kurum içi iş performansının, verimliliğin ve müşteri memnuniyetinin arttırılması ve istenmeyen çağrıların engellenmesini sağlayacak telefon uygulamaları olarak sıralanmaktadır. Tasarlanan sistemin bu senaryolara uygulanması sonucu

çalışan motivasyonunu düşürmeden kurum performansı ve verimliliği açısından önemli artılar elde edilebileceğinin yanı sıra telefon maliyetlerinde de ciddi oranlarda tasarruf sağlanabileceği görülmüştür.

Yapılan tez çalışmasının özgün değeri ve hedeflenen yaygın etkisi ile bu alanda bugüne kadar öne sürülen fikirler ve yapılan çalışmalar hakkında daha ayrıntılı bilgi ikinci bölümde verilmektedir. Tasarlanan yapının daha iyi anlaşılabilmesi için üçüncü bölümde sesli iletişim teknolojileri ile ilgili ayrıntılı bilgi verilmiştir. Dördüncü bölümde iletişim denetleme ve kontrol sisteminin tasarım aşaması, beşinci bölümde uygulamanın gerçeklenmesi anlatılmaktadır. Altıncı bölümde gerçeklenen paket ve sınıflar hakkında ayrıntılı bilgi ve sınıf diyagramları yer almaktadır. Yedinci bölümde sistemin uygulandığı senaryolar ele alınmış, sekizinci ve son bölümde ise sonuçlar değerlendirilmiştir.

2. ÖZGÜN DEĞER VE YAYGIN ETKİ

Bilgi sistemlerindeki gelişmeler, ticari kuruluşlardaki bir çok iş sürecinin kalkmasına veya otomatikleşmesini sağlasa da, günümüz iş dünyası göz önüne alındığında, özellikle hizmet sektörü başta olmak üzere, telefonla iş geliştiren kişilerin mesailerinin büyük bir kısmı telefonla konuşarak harcanmaktadır. Gelişen teknolojilerin sağladığı yeni kaynaklar, kişilere iş konuşabilecekleri yeni zamanlar kazandırmaktadır. Bu da iş hayatındaki sesli iletişim hacminin gittikçe artmasına neden olmaktadır.

İşletmelerdeki sesli iletişim süreçleri incelendiğinde, çeşitli eksiklikler ve yaygın uygulama hataları tespit edilmektedir. Özellikle yoğun telefon trafiği olan bir ortamda, yanlış bağlanan telefonlar, gereksiz uzatılan diyaloglar ve istenmeyen aramalar, çalışan performansı üzerinde önemli negatif etkiler göstermektedir. Benzer şekilde, müşterilerin işletmeyi aradığında doğru kişiye ulaşana kadar geçen süre zarfında karşılaşabilecekleri gecikme ve zorluklar, müşteri memnuniyeti ve kurum performansı açısından da önemli bir parametre olmaktadır.

İşletmelerdeki bir diğer önemli sorun da, sesli iletişim kaynaklarının iş süreçleri dışındaki kullanımıdır. Bu kullanım sonucu ortaya çıkan ek maddi yükün yanı sıra yaşanan mesai kayıpları, genel kurum performansını olumsuz etkilemektedir. Kurumdaki kişisel telefon trafiğini önleyici faaliyetler ise motivasyon ve iş verimliliğinin azalmasına neden olmaktadır. Bu yüzden çalışanlar üzerinde baskı ve rahatsızlık oluşturmayacak çözümlere ihtiyaç duyulmaktadır.

İşletmelerdeki telefon performansını, dolayısıyla iş verimini etkileyebilecek bir diğer faktör de SPIT (Spam Over Internet Telephony) faaliyetleridir. Geliştirilen robot operatörler sayesinde aynı anda binlerce telefon aranarak reklam kayıtları dinletilmektedir. Bu operasyonların düşen maliyetler sayesinde giderek daha da artmasıyla şirketlerin çeşitli önleyici çözümlere yönelmesi bir ihtiyaç haline gelmiştir.

Daha önce kapsamlı olarak ele alınmamış bir alanda gerçekleştirilen bu tez çalışması, kurumların sesli iletişim kaynakları yönetimini ele alarak, mevcut ve gelecekteki ihtiyaçlarına yönelik önemli artılar ortaya koymuştur. Ayrıca, işletmelerin telefon iletişimini yöneterek başta iş performansı olmak üzere, müşteri memnuniyeti ve çalışan motivasyonu gibi önemli kalite kriterlerine katkıda bulunmakta, işletmenin varolan iletişim kaynaklarının daha verimli kullanılmasını sağlayarak iletişim giderlerinin azalmasını sağlayabilmektedir.

Mevcut potansiyele katkıda bulunmanın yanı sıra, kurumun gelecekte karşılaşabileceği istenmeyen faaliyetler de, bu çalışmanın ortaya koyduğu yapı sayesinde önlenebilmektedir.

Bu koruyucu fonksiyonlar sayesinde işletmenin sahip olduğu sesli iletişim kaynaklarından daha yüksek oranda yararlanması sağlanmıştır.

Tez çalışması kapsamında gerçekleştirilmiş sistem, marka ve platformdan bağımsız olarak gerçeklendiğinden, telefonu iş süreçlerinde yoğun olarak kullanan fakat farklı alt yapı ve sistemlere sahip işletmelerde kullanılabilecektir. Sistem ölçeklenebilir bir tasarım temel alınarak gerçeklendiğinden her ölçekteki işletme, uygun uygulama maliyetleri dahilinde ihtiyacına yönelik sistemi kurabilecektir.

Uygulamanın kullanımı sonucu elde edilecek verilerle kurum içi kullanım alışkanlıklarının ve faaliyetlerinin belirlenmesi, sınıflandırılması ve incelenmesi gibi faaliyetlerin de sağlanması için önemli sonuçlar ortaya konabilmektedir.

Literatürde sesli iletişimin denetlenmesi ve yönetilmesi vasıtasıyla iş verimin arttırılmasına yönelik detaylı bir araştırma yer almamaktadır. Bugüne kadar iletişim alanında yapılan akademik çalışmalar çoğunlukla iletişim alt yapılarının ve iletişim protokollerinin geliştirilmesine ve iyileştirmesine yönelik olmuştur.

İstenmeyen telefon trafiğinin belirlenmesine yönelik çalışmalarında Robert MacIntosh ve Dimitri Vinokurov, VoIP iletişimde SPAM tespitine yönelik araştırmalar yapmıştır (MacIntosh R., 2005). Bu çalışma, sinyalleşme protokolünü analiz ederek servis sağlayıcıların istenmeyen içeriği engelleyebileceği bir yöntem ortaya koymayı amaçlamaktadır. Fakat bu araştırma, sesli iletişimin tek bir alanına yönelik bir çalışma olması, yalnız servis sağlayıcı tarafında gerçekleştirilebilmesi ve istenmeyen trafiğin önlenmesine yönelik kontrol fonksiyonlarını sağlamaması açısından varolan ihtiyacı karşılayacak bir çözüm ortaya koyamamıştır.

Cisco [5], Alcatel-Lucent [6] ve Karel [7] gibi bazı önemli santral üreticileri, kendi ürünlerine yönelik bir takım kontrol ve izleme mekanizmaları geliştirseler de, bu özellikleri sunan ürünler yüksek maliyetleri ve sadece numara veya yöne bağlı yönlendirme yapabilmek gibi sınırlı fonksiyonlarıyla yetersiz kalmaktadır.

Kenneth J. Turner, Stephan Reiff-Marganiec, Lynne Blair, Jianxiong Pang, Tom Gray, Peter Perry ve Joe Ireland'dan oluşan çalışma grubu, bu alandaki eksiği görmüş ve yaptıkları araştırma ile ileride geliştirilebilecek genel amaçlı bir çağrı kontrol sistemi için bir dil ve kurallar sistemi hazırlamışlardır (Turner K.J., 2005). Fakat bu tarz bir sistemi, marka ve altyapıdan bağımsız olarak gerçekleyecek bir uygulama henüz ortaya koyulmamıştır.

Sesli haberleşmede insan ve donanım faktörüne yönelik araştırmalar da literatürde kendine yer bulmuştur. Jevtic, yaptığı çalışmada [Jevtic D, 1998], dört aylık yerel telefon trafiğini izleyerek, insanların telefonla iletişim alışkanlıklarına yönelik tespitlerde bulunmuş, başarısız telefon trafiğinin, servis kalitesi ve müşteri memnuniyeti parametreleriyle olan ilişkisini ortaya koymaya çalışmıştır. Bir diğer çalışmada, Xiaotao Wu ve Henning Schulzrinne, karar ağaçları gibi yöntemlerle gerçekleştirilmiş servis uygulamalarının Internet telefonu hizmetlerinde ortaya koyacağı risk üzerine araştırmalarda bulunmuştur (Wu X., 2005).

Bütün bu bilgiler ışığında, bu tez çalışması sonucunda gerçeklenen uygulama, yukarıda belirtilen ihtiyaçlara ve yapılan araştırmaların eksik bıraktığı alanlara cevap verebilecek şekilde, altyapıdan bağımsız olarak, kullanılan santral veya VoIP Gateway ile koordinasyon içinde çalışarak, mevcut sesli iletişim trafiğini izlemeye ve önceden belirlenmiş kurallar doğrultusunda, bu trafikte düzenlemeler yapmaya olanak sağlamaktadır.

Düzenleme faaliyetleri, görüşmeyi kaynak veya hedefe bağlı olarak, otomatik aktarma, kesme ve görüşme iznini belirleme gibi çeşitli fonksiyonları kapsamaktadır. Bu kontrol fonksiyonları, görüşme zamanı, süresi, hedef-kaynak, arama tipi gibi organizasyon yetkilileri tarafından belirlenebilecek ön tanımlı parametrelere bağlanabileceği gibi, ihtiyaçlar ve tercihler doğrultusunda organizasyonun belirli bir zaman zarfı içindeki telefon trafiğinden elde edilen veriyi girdi olarak kullanan bir uzman sistem ile dinamik olarak da gerçekleştirilebilmektedir.

Bu tez çalışması ile, istenmeyen telefon trafiğinin engellenmesinin yanı sıra, çeşitli telefon aktarma-bağlama aksaklıklarının da önüne geçilmesi sağlanarak organizasyondaki verime katkıda bulunmak amaçlanmıştır.

3. SESLİ İLETİŞİM

Ayrı lokasyonlarda bulunan kişilerin birbirleriyle konuşmasını veya farklı noktalardaki ses bilgisinin diğerine aktarılmasını sağlamak amacıyla başta telefonlar olmak üzere çeşitli iletişim cihazlarının kullanımına sesli iletişim denilmektedir.

İlk ortaya çıktığında telefonlar, birbiriyle ikili oluşturacak şekilde direk bağlanıyor farklı bir telefona erişmek istenildiğinde aradaki bağlantı kablosunun değiştirilmesi gerekiyordu. Fakat, telefon kullanıcısı sayısının artması bu yöntemi kullanışsız kılmış ve santral kavramının ortaya atılmasına neden olmuştur. Santraller sayesinde telefonları kaynak noktada ikili olarak eşlemek yerine tek bir noktada birleştirerek istenilen çiftin birbiriyle görüşmesine imkan tanınmıştır.

Günümüzde telefonlar bir kablo vasıtasıyla yerel bir santrale bağlanmakta, bir sonraki aşamada da bu santrallerin kendi aralarında bağlanılmasıyla açık anahtarlamalı telefon ağı (Public Switched Telephone Network, PSTN) oluşmaktadır. Özellikle son dönemde bu yapı daha da geliştirilmiş, ara bağlantıların büyük ölçüde sayısal hale getirilmesi sağlanmış, bunun sonucunda da maliyetlerin düşürülmesi ve ses kalitesinin arttırılmasının yanı sıra yeni ağ ve veri aktarım hizmetlerinin de verilebilmesine imkan tanınmıştır.

Sesli iletişim alt yapısındaki sayısal gelişim ve veri aktarımının da bu altyapı üzerinden sağlanabilmesi, İletim kontrol protokolü / Internet Protokolü (Transmission Control Protocol / IP Protocol, TCP/IP) [8] üzerinden telefon hizmetlerinin verildiği yeni bir yapının ortaya çıkmasına neden olmuştur. IP üzerinden ses iletimi (Voice over IP, VoIP) adı verilen bu yeni yapı ile, hem daha kaliteli bir sesli iletişim mümkün olmuş, hem de geleneksel sesli iletişime yepyeni iletişim fonksiyonlarının kazandırılması sağlanmıştır.

Sesli iletişim cihazlarındaki gelişme ile birlikte, bu cihazların bilgisayarlarla olan entegrasyonu ve ses iletişiminin bilgisayarlar aracılığıyla izlenebilmesi mümkün olmuştur. Bilgisayar telefon entegrasyonu (Computer Telephony Integration, CTI) adı verilen bu kavram yeni ortaya atılmamakla birlikte geçmişte sadece önemli çağrı merkezlerinde maliyet hesaplama veya otomatik çağrı dağıtımı gibi çağrı merkezlerine özel fonksiyonları gerçekleştirmek amacıyla ele alınmaktaydı. Fakat gelişen teknolojinin getirdiği yeni nesil araç ve uygulamalar sayesinde bu kavram, telefonu etkin olarak kullanan tüm kurum ve kişilere yönelik fonksiyonları sunabilmek adına daha yaygın ve önemli hale gelmektedir.

3.1 Açık Anahtarlamalı Telefon Ağı (PSTN)

Açık anahtarlamalı telefon ağı, dünyadaki devre anahtarlamalı telefon ağlarının bir araya geldiği bir yapıdır. Son dönemde sabit telefonların yanı sıra mobil telefon ağları da bu sisteme dahil edilmiştir.

Açık anahtarlamalı telefon ağı, Uluslararası Telekomünikasyon Birliğinin (ITU) belirlediği teknik standartlar çevresinde idare edilmekte ve telefon numarası olarak da bilinen E.164 [9] adresleme yapısını kullanmaktadır.

Açık anahtarlamalı telefon ağı, servis kalitesi özelliklerini sağlayan ilk trafik mühendisliği çalışması olarak kabul edilir. A.K. Erlang (1878-1929) tarafından ortaya atılan metodlar sayesinde belirli bir seviyede servis verebilmek için gerekli teçhizat ve personel sayısı belirlenebilmektedir. (Erlang A.K.,1909).

Açık anahtarlamalı telefon ağına doğrudan bağlı olmayan birçok özel telefon ağı da bulunmaktadır. Bunlar başta askeri ve ticari olmak üzere farklı amaçlarla, büyük kurumlar tarafından iç iletişimde kullanılmakta, sadece gerekli noktalarda kurumum yönettiği özel santraller (Private Branch eXchange, PBX) üzerinden ana telefon ağına bağlantı sağlanmaktadır.

3.1.1 Telefon Numaraları

Telefon numarası, telefon ağında bir son noktayı belirten ve ondalık rakamlardan oluşan bir sayı dizisidir. Bu numara bir telefon çağrısının yapılması için gereken bilgiyi içermektedir.

Açık anahtarlamalı telefon ağına dahil olan ülkelerde genellikle bir nokta, özel bir hizmet ya da amaç doğrultusunda ayrılmadığı sürece belirli uzunluktaki bir telefon numarasına sahip olmaktadır. Acil durum servisleri gibi kısa numaraların kullanılması gereken durumlarda, bu numaraların orjinal hali olan uzun ve tekil haline çevrimi, sesli iletişim hizmetini sağlayan kurum tarafından otomatik olarak gerçekleştirilir.

Uluslararası telefon ağında numaralandırma için Uluslararası Telekomünikasyon Birliği'nin önerdiği E.164 adreslemesi kullanılmaktadır. Buna göre Dünya üzerindeki coğrafi bir alanda kullanılabilecek numaralar için çizelge 3.1'de gösterilen şekilde 1-3 basamaklı bir ülke kodu ve en fazla 12 basamaklı bir abone numaradan oluşan yapı kullanılmaktadır.

Çizelge 3.1 E.164 adreslemesine göre genel telefon numara yapısı

Ülke Kodu	**Küresel Abone Numarası**
1-3 basamak	En fazla 12 basamak
Toplamda en fazla 15 basamak	

Daha özel olarak, eğer karşılanan coğrafi bölge, daha alt alanlara bölünüyorsa, telefon numaraları çizelge 3.2'de görüldüğü üzere 1-3 basamaklı bir ülke kodu ile başlamakta, bunu 1-4 basamaklı bir alan kodu izlemekte ve en son olarak da abone numarasının eklenmesiyle toplamda 15 basamağı geçmeyecek bir yapı oluşturmaktadır.

Çizelge 3.2 E.164 adreslemesine göre telefon ağı numara yapısı

Ülke Kodu	**Grup/Alan Kodu (=X)**	**Abone Numarası**
1-3 basamak	1-4 basamak	En fazla 12-X basamak
Toplamda en fazla 15 basamak		

Yine Uluslararası Telekomünikasyon Birliği'nin E.123 [10] numaralı tavsiyesine göre uluslarası bir numara belirtilirken ülke kodundan önce artı (+) işareti gelmeli ve yine bu tarz bir numara aranmadan önce bu işarete karşılık gelen ve her ülkenin kendi belirleyeceği bir numara, bu işaretin yerine kullanılmalıdır. Sabit telefonların aksine mobil telefonlarda artı işareti doğrudan kullanılabilmektedir.

Kurumların kendi özel telefon ağlarına yönelik numaralandırma için belirli bir standart olmamakla birlikte kurumun ölçeğine bağlı olarak genellikle 3 veya 4 basamaklı numaralar tercih edilmektedir.

Açık anahtarlamalı telefon ağında bölgesel ve yerel numaraların formatı, atanması ve idaresi, her ülkenin kendi belirlediği yetkili makamlara bırakılmıştır. Özel telefon ağlarında ise bu görev genellikle diğer iletişim alt yapısı ile beraber bilgi işlem yetkililerince yürütülmektedir.

3.2 IP Üzerinden Ses İletimi (VoIP)

IP Üzerinden Ses İletimi (VoIP), sesli iletişimin geleneksel telefon ağı yerine Internet veya IP temelli herhangi bir ağ üzerinden taşınması ve yönlendirilmesidir. Internet telefon ağı olarak da bilinen bu sistemde, kullanıcılar Internet telefonu olarak adlandırılan uç noktalarda ses

bilgisini IP üzerinden göndermeye ve almaya yaran cihazları kullanarak mevcut IP ağı üzerinden sesli iletişim gerçekleştirebilmektedir. Internet telefonları arasındaki görüşmeler, açık anahtarlamalı telefon ağından bağımsız mevcut veri altyapısı üzerinden yapıldığından görüşme için bir ücret ödenmesine gerek kalmamakta, sadece sistem dışındaki telefonları aramak üzere açık anahtarlamalı telefon ağına bağlantı sağlayan ağ geçitlerinin kullanımında belirli bir maliyet oluşmaktadır.

IP üzerinden ses iletiminin, geleneksel telefon sistemine göre başlıca avantajları aşağıda sıralanmıştır:

- Tek bir telefona birden fazla hat/numara verilebilmesi: Bu sayede tek bir lokasyona yeni bir hat eklenmesi oldukça kolaylaşmaktadır.
- Telefonun fiziksel lokasyonunun değiştirildiğinde dahi aynı hattın/numaranın kullanılabilmesi.
- Geleneksel telefon sistemlerinde servis sağlayıcıların numara yönlendirme, telekonferans gibi ek ücretle verdiği bazı hizmetlerin standart ve ücretsiz olarak kullanılabilmesi.
- Internet üzerindeki video, mesajlaşma, dosya transferi gibi diğer servislerle entegrasyon sağlanarak daha bütünleşik bir iletişim yapısının oluşturması.
- Farklı ihtiyaçlara yönelik uygulamalar geliştirerek yeni fonksiyonların hızlı ve kolayca eklenebilmesi.

IP üzerinden ses iletiminin sağladığı ve yukarıdaki maddelerde belirtilen avantajlardan yararlanılabilmesi için aşağıda listelenen noktaların özellikle dikkate alınması gerekmektedir:

- Bant Genişliği: Veri alt yapısındaki bant genişliği, telefon trafiğini taşımak için yeterli seviyede olmalıdır.
- Gecikme Süresi: Gecikme süresi ve paket kaybı değerlerinin yüksek olması durumunda, ses kalitesinde düşüş ya da ses kaybı yaşanacağından iletişim mümkün olmayacaktır.
- Sistemin Kararlılığı: Kullanılan sesli iletişim ve ağ cihazların yedekli olması; sinyal kalitesini yüksek, parazit seviyesini düşük tutacak yapıların kurulması sağlıklı ve kesintisiz ses iletişimi için çok önemlidir.
- Güvenlik: Gerçekleştirilen sesli görüşmenin istenmeyen kişiler tarafından dinlenememesi için şifreleme gibi yöntemlerin ele alınması çağrının gizliliği açısından önemlidir.

3.3 Telefon Santrali

Telefon santrali, telefon çağrılarını bağlayan elektronik ve mekanik bileşenlerden oluşan bir sistemdir. Günümüzde santraller, merkezi ve yerel olmak üzere farklı seviyelerde çoğunlukla sayısal bağlantılar kullanılarak hizmet vermektedirler.

İlk santraller, artan kullanıcı sayısıyla beraber telefonların eşlenmiş çiftler olarak bağlanmasının farklı telefonlarla görüşme ihtiyacını karşılayamaması sonucu ortaya çıkmıştır.

Bu ilk santrallerde, görevli operatörler kaynak ve hedef kullanıcıların hatlarını bağlayacak anahtarlamayı elle yapmakta, yeni bir eşleşme gerektiğinde gerekli değişikliği yine elle gerçekleştirmekteydi. Eğer görüşmenin yapılacağı kullanıcı farklı bir santralde yer alıyorsa, operatörler birbirleriyle iletişim kurarak önce santraller arası bağlantının kurulmasını, daha sonra da kullanıcıların bağlanmasını sağlamaktaydı.

1900'lü yılların başıyla birlikte santral operatörlerin yerini elektromekanik sistemler almaya başlamıştır. Bu sistemlerle birlikte telefonun açıldığı otomatik olarak belirlenmekte, çevrilen numaraya göre yerel kullanıcı ya da uzak santral bağlantısı operatöre gerek kalmadan yapılmaktaydı. Bu sistemlerle birlikte ücretlendirme sistemleri de kullanılmaya başlanmış, eskiden telefonu bağlayan operatörün tuttuğu kayıtların da bu sistemler tarafından alınması sağlanmıştır.

Günümüzde santraller büyük ölçüde sayısallaşmıştır. Ses sinyalinin kullanıcıdan santrale gelmesiyle birlikte sinyal sayısal veriye çevrilir ve bundan sonraki tüm anahtarlama ve taşıma işlemleri sayısal ortamda gerçekleşir. Bilgi, hedefe ulaştırılmadan önce tekrar ses sinyaline çevrilir ve daha sonra kullanıcıya iletilir. Tüm bu işlemler belirli bir gecikme ortaya koymakla birlikte, servis kalitesi ilkeleri doğrultusunda yapılan ağ mühendisliği çalışmaları ile bu gecikmenin minimum düzeyde tutulmasına çalışılmaktadır.

Günümüzde kullanılan santraller yedekli ve hata toleranslı olarak hizmet vermektedir. Eğer alt seviye santrallerden birinde sorun oluşursa, kontrol sistemleri hemen santral üzerindeki hatları kullanılıyor olarak işaretler ve yeni bağlantıların bu santral üzerinden yapılmasını engeller. Bir süre sonra, santral üzerindeki aktif bağlantıların sona ermesiyle birlikte gerekli bakım ve onarım çalışmalarının başlaması sağlanır. Yine sorunlu bir bağlantının tekrar etmemesi için hatların atanmasında ilk gelen ilk çıkar (First In-First Out, FIFO) prensibi kullanılır. Buna göre bir bağlantıda sorun veya parazit oluşmuşsa, aynı bağlantı tekrar edildiğinde bu sefer başka bir hattın tahsis edilmesi sağlanır.

3.3.1 Özel Santral (Private Branch eXchange, PBX)

PBX, bir kurumdaki telefonların birbirlerini ya da açık anahtarlamalı telefon ağındaki bir telefonu aramasını sağlayan yerel bir santraldir.

Aynı PBX altındaki telefonlar, ek bir maliyet ödemeden birbirini arayabilmektedir. Ayrıca kullanılan cihazın marka ve modeline göre geleneksel telefon sistemindeki santrallerde olmayan bir takım ek özellikler kullanıcılara sunulabilmektedir.

PBX'e bağlı telefonlara verilen numaralar dahili numara olarak adlandırılır. PBX'in ana telefon şebekesine bağlı hatlarına ise dış hat denilmektedir. Dahili telefonlar, dış hatları kullanarak istedikleri bir numarayı arayabilmektedir. Fakat aynı anda dış numaralarla yapılabilecek görüşme sayısı dış hat sayısını geçmemelidir. Aynı anda yapılabilecek dahili görüşme sayısı ise kullanılan PBX'in model ve konfigürasyonuna göre değişiklik gösterebilir.

Gelişmiş PBX sistemleri, aranan numaraya göre dış hat seçebilmektedir. En düşük maliyetli yönlendirme (Least Cost Routing – LCR) adı verilen bu sistem sayesinde boş olan ilk dış hattı kullanmak yerine farklı ücret tarifelerine sahip dış hatlardan en ucuz maliyet getirecek olanı arama anında seçilebilmektedir. Yine marka ve modele bağlı olarak çağrı dağıtma, çağrı yönlendirme, çağrı bekletme, otomatik geri arama, telekonferans, direk dahili arama, telesekreter servisi gibi özellikler de güncel PBX'lerde yer almaktadır.

3.3.1.1 Yazılımsal Anahtarlama (Soft Switch)

Yazılımsal anahtarlama, tüm PBX işlevlerinin bilgisayar üzerindeki bir yazılım tarafından gerçekleştirildiği telefon ağı cihazıdır. Sistem sesli iletişim cihazlarının ve hatların bağlanabileceği kartları içeren bir bilgisayardan ve ses iletişim yönetimi yazılımından oluşmaktadır.

Sistemdeki çağrı kontrol bileşeni, ücretlendirme, yönlendirme ve çağrı servisleri gibi mantıksal işlemleri yerine getirirken, medya geçidi bileşeni farklı tipteki bağlantıların bir araya getirilmesi ve veri aktarımı gibi işlemleri karşılar.

Genellikle PBX'lere göre daha az maliyetli olan yazılımsal anahtarlamalar, donanımsal PBX'ler kadar yüksek performans göstermediğinden, çoğunlukla kurumların dahil olduğu farklı telefon ağları arası geçiş görevini yerine getirmek ya da mevcut PBX'e ek özellikler sağlamak üzere kullanılmaktadır.

Bu tez kapsamında açık kaynak kodlu bir yazılımsal anahtarlama uygulaması olan Asterisk kullanılmıştır. Asterisk, Linux [11], Mac Os X [12], OpenBSD [13], FreeBSD [14] ve Sun Solaris [15] gibi farklı platformlarda çalışabilmekte, VoIP başta olmak üzere farklı sesli iletişim protokollerini desteklemektedir. Açık lisanslama (General Public Licence, GPL) ile ücretsiz olarak dağıtılan Asterisk, dünya çapında farklı ülkelerden birçok uygulama geliştirici tarafından desteklenmektedir.

3.3.1.2 iPBX

iPBX, geleneksel PBX sisteminin Internet telefonları için kullanulan IP temelli versiyonudur. Bu sayede, PBX üzerindeki servislere, Internet üzerinden ulaşılabilmekte, kullanıcıların PBX ile aynı fiziksel lokasyonda bulunması gerekmemektedir.

Internet telefonu sistemleri için farklı markaların sunduğu birçok iPBX bulunmaktadır. Bunlar genellikle kendi standart ve protokollerini kullansa da, birbirleriyle iletişim kurmak için belirli ortak standartlar da oluşturulmuştur. Gerektiği durumlarda kurulan ağ geçidi sistemleriyle farklı alt yapılara sahip ağlar arası geçiş sağlanabilmekte, farklı iPBX'ler ve geleneksel PBX'ler arasında istenilen sesli iletişim sağlanabilmektedir.

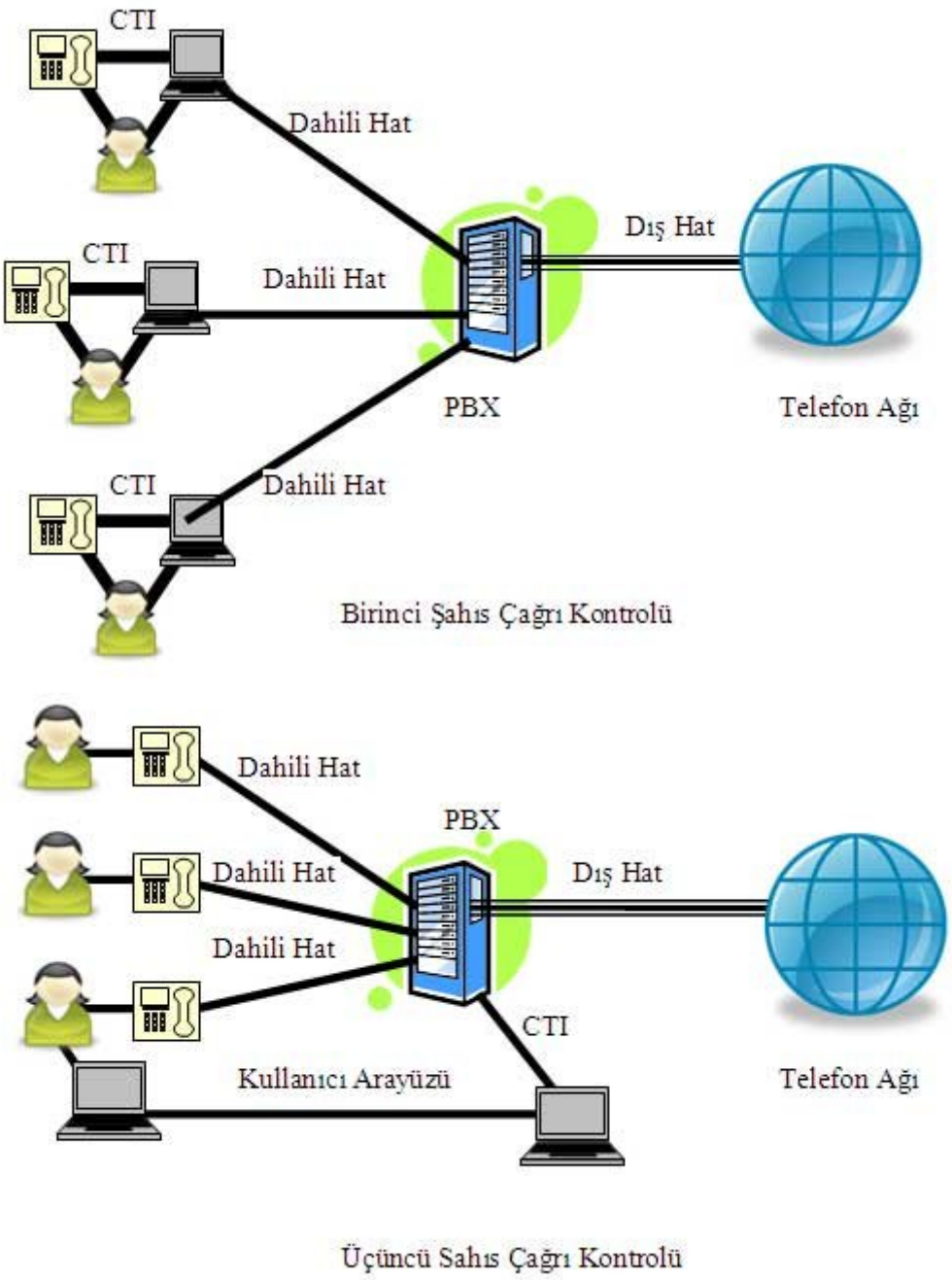
Bu sistemlerin hizmet verebilmesi için yeterli bant genişliğine sahip olması ve Internet bağlantısının güvenirliğinin garanti edilebilmesi çoğu zaman yeterli olmaktadır.

3.4 Bilgisayar Telefon Entegrasyonu (Computer Telephony Integration, CTI)

Bilgisayar telefon entegrasyonu (CTI), bir kurumdaki telefon ve bilgisayar sistemlerinin bütünleştirilmesi ve koordine edilmesi teknolojisidir. Özellikle iletişim parametrelerinin artması ve çeşitlilik kazanmasıyla birlikte sesli iletişim sistemlerine yeni özellikler kazandırmak adına, bilgisayarların kullanılması önemli bir ihtiyaç haline gelmiştir. Günümüzde CTI kullanılarak şu gibi fonksiyonlar gerçeklenebilmektedir:

- Çağrı bilgisinin görüntülenmesi
- Otomatik arama
- Otomatik yönlendirme
- Çağrı kontrolü
- Farklı çağrılar arası koordinasyon
- Çağrı merkezi fonksiyonları
- Telefonla başka sistemlerin kontrol edilmesi

CTI uygulamaları şekil 3.1'de görüldüğü gibi iki tipte gerçekleştirilmektedir: Birinci şahıs çağrı kontolü (First Party Call Control), Üçüncü şahıs çağrı kontrolü (Third Party Call Control).



Şekil 3.1 Birinci şahıs çağrı kontrolü ve üçüncü şahıs çağrı kontrolü

Birinci şahıs çağrı kontrolünde, kullanıcının bilgisayarı ve telefonu arasında direk bağlantı bulunmaktadır. Bilgisayardaki modem üzerinden yapılan görüşmeler üzerinde gerçekleştirilebilecek uygulamalar bu gruba dahil olmaktadır. Birinci şahıs çağrı kontrol yazılımları, geliştirilmeleri kolay olmakla birlikte büyük ölçekteki uygulamalar için uygun olmamaktadırlar.

Üçüncü şahıs çağrı kontrolünde, uygulama PBX ile bağantı kurar ve gerekli işlemleri direk bu

bağlantı üzerinden gerçekleştirir. Uygulama gerektiği durumlarda telefon sunucusunun yanı sıra kullanıcı bilgisayarlarıyla da iletişim kurabilir. Çağrı merkezlerinde arayan kişinin bilgilerinin, çağrıya cevap veren kullanıcının bilgisayar ekranına getirilmesi bu tarz bir uygulamadır. Benzer şekilde kullanıcı, bilgisayarından CTI uygulamasına bir komut göndererek istediği bir numaranın aranmasını da sağlayabilir. Üçüncü şahıs çağrı kontrolü uygulamalarında kullanıcının telefonunun bilgisayara doğrudan bağlı olması gerekmediğinden, çok sayıda telefonun ve çağrının bulunduğu büyük ölçekli kurumlarda, bu tipteki uygulamaların gerçeklenmesi daha kolay olmaktadır.

Her markanın kendi protokolünü oluşturduğu eski dönemlerle karşılaştırıldığında, günümüz CTI uygulamalarının standartlaştırılması yolunda önemli ilerleme sağlanmıştır. Uluslararası Telekomünikasyon Birliğinin onayladığı bilgisayar destekli telekomünikasyon uygulamaları (Comuter-Supported Telecommunication Applications, CSTA) [16], AT&T [17] ve Novell [18] şirketleri tarafından desteklenen TSAPI [19], Sun Microsystems [20] tarafından desteklenen JTAPI [21] ve Microsoft [22] tarafından desteklenen TAPI[23], başlıca CTI uygulama platformları olarak kullanılmaktadır.

4. TASARIM

Bu tez çalışmasında, bu konuyla ilgili daha önceden yapılmış ve ikinci bölümde açıklanan çalışmalarda ele alınmamış veya eksik bırakılmış yönler ön plana çıkarılarak, aşağıdaki açıklanan temel özellikler çerçevesinde yeni bir sistem tasarlanmıştır.

Sistem, iş süreçlerinde sesli iletişimi yoğun olarak kullanan orta ve büyük ölçekli işletmeler için uygun olmalıdır. İşletmenin büyüklüğüne ve telefon trafiği yoğunluğuna bağlı olarak sistemin ölçeklendirilmesi gerçekleştirilebilmelidir. Böylece her ölçekteki işletmelere uygun bir çözüm ortaya konarak, santral firmalarının sunduğu yetersiz ve yüksek maliyetli sistemlere bir alternatif sağlanmış olacaktır.

Gerçeklenen uygulama, analog, sayısal ve VoIP haberleşme altyapısını kullanabilmelidir. Bu altyapıları destekleyen santral marka ve modelleri ile tam uyumluluk esas alınmalıdır. Bu sayede, sesli haberleşme için marka ve modelden bağımsız, genel bir iletişim yönetimi çözümü ortaya konulabilecektir.

Sistem, sesli iletişimin yönetimi için gerekli temel fonksiyon ve özellikleri içinde barındırmalıdır. Bunları kısaca gruplamak gerekirse:

- **Gelen Telefonların Engellenmesi:** Sistemde, statik veya dinamik listeler tanımlanarak, bu çağrıların engellenmesi ve kullanıcılara bağlanmaması sağlanabilmelidir. Engelleme, geciktirerek aktarma, sesli posta sistemine yönlendirme veya tamamen çağrının düşürülmesi şeklinde olabilir. Statik listeler, doğrudan tanımlanmış hedef, kaynak ve koşulları içerirken, dinamik listeler, mevcut telefon trafiğinden elde edilen veriler kullanılarak sistem tarafından otomatik kurallar tanımlanmasıyla oluşturulabilir. Örneğin telefon trafiğini engelleyecek biçimde çağrı gerçekleştiren bir numaranın bloke edilmesi, otomatik olarak gerçekleştirilebilir.

- **Gelen Telefonları Yönlendirme:** Gelen telefonların engellenmesi yerine, başka bir kanala yönlendirilmesi gerekli olduğunda, sistem bunu kendiliğinden gerçekleştirebilmelidir. Örneğin kredisi dolan ve ödemelerini geciktiren bir müşteri yeni bir sipariş daha vermek için satış bölümünü aradığında, doğrudan tahsilat bölümüne bağlanabilir veya telefonda ücretli yardım masası hizmeti veren bir şirket, anlaşmalı müşterileri aradığında yardım masasına, anlaşmalı olmayan müşterileri aradığında ise satış departmanın yönlendirme yapabilir. Müşteri temsilciliği yapısında çalışan bir şirketi arayan bir müşterinin, doğrudan kendi müşteri temsilcisine bağlanması sağlanabilir. Bu gibi durumları karşılayabilecek tanımlamaların

yapılabileceği gelişmiş bir yönlendirme sistemi uygulamada bulunmalı, kurum performansına önemli bir katkıda bulunacak bu işlevlerin yerine getirilmesi sağlanabilmelidir.

Telefon yönlendirmesinin bir başka şekli de, kısa zaman aralıkları ile aynı kaynaktan gelen çağrıların aynı kişiler tarafından yanıtlanmasını sağlamak olmalıdır. Böylece arayan kişilerin, telefon santralinde dakikalar kaybetmeden doğrudan ilgili insanlara ulaşması sağlanarak, proje hedeflerinden biri olan müşteri memnuniyetinin arttırılmasına katkıda bulunulacaktır.

- **Giden Çağrıların Denetlenmesi ve Engellenmesi:** Bu sistem sayesinde çalışanların istenmeyen telefon aramalarını yapmaları denetlenebilmeli ve gerekli görülürse engellenebilmelidir. Örneğin bir mağazadaki personelin, sadece müşteri veritabanında bulunan telefon numaralarını arayabilmeleri, başka numaraları arayamamaları sağlanabilmelidir. Bu sayede telefon kaynaklarında tasarruf elde edilmiş olacaktır.

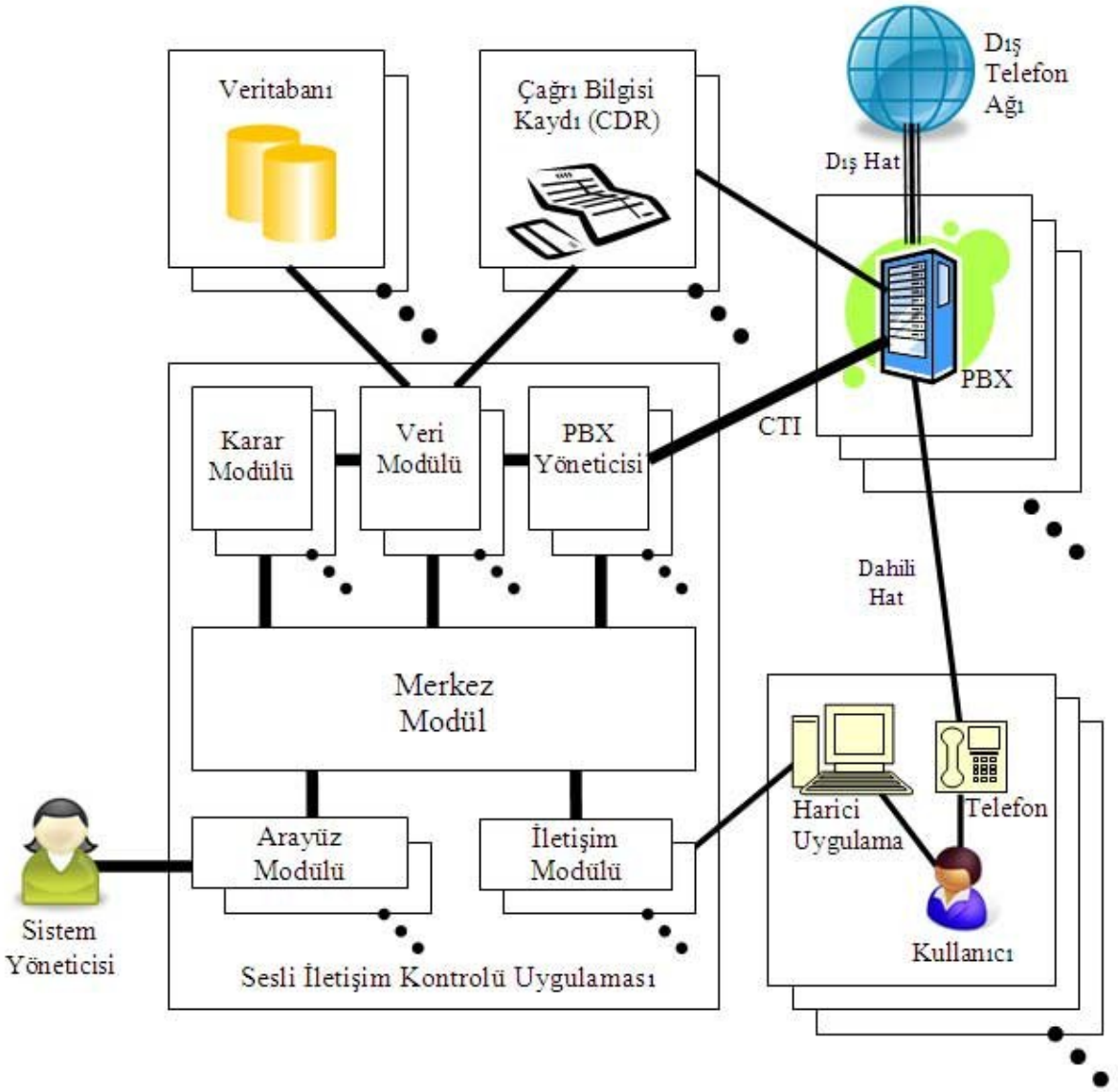
Sistem, sunduğu işlevlerin kolayca kullanılabileceği, gerekli tanımların hızlı ve anlaşılır bir şekilde yapılabileceği özellikte olmalıdır. Bu sayede mevcut sistemlerin karmaşık fakat yetersiz kullanımına bir alternatif ortaya konulmuş olacaktır.

Tasarlanan sistem, sesli iletişim sistemlerindeki kabul edilebilir seviyelerin üstünde gecikme süreleri oluşturmayacak yapıda olmalıdır. Günümüz sesli iletişim sistemlerinde, bir çağrının beklenen başlatılma süresi, çağrının arama yönüne bağlı olmak üzere 1 ile 15 saniye arasında değişmektedir. Buna göre darboğaz oluşabilecek modüllerin paralel olarak çalışabilmesi sağlanarak görüşmenin beklenen süreler dahilinde başlatılması sağlanmalıdır.

Yukarıdaki özellikleri sağlayacak şekilde bir tasarım temel alınarak Şekil 4.1'de görüldüğü gibi 6 modül oluşturulmuştur. Bu modüller,

- Merkez Modül
- PBX Yöneticisi
- Karar Modülü
- Veri Modülü
- İletişim Modülü
- Arayüz Modülü

olarak isimlendirilmiştir.



Şekil 4.1 Sesli iletişim kontrol sistemi tasarımı

4.1 Merkez Modül

Merkez modül sistemde yer alan diğer modülleri başlatmak, yönetmek ve aralarındaki iletişimi sağlamakla yükümlüdür. Buna göre bu modül tarafından gerçekleştirilen başlıca işlemler,

- Sistem tanımlarının veya kullanıcı komutlarının gerektirdiği şekilde diğer modülleri başlatmak veya sonlandırmak
- PBX yöneticisinden gelen olayları uygun bir karar modülüne iletmek
- Karar modülü tarafından verilen komutları ilgili PBX modülüne iletmek
- Arayüz ve iletişim modülüne, sistemin durumunu ve gerçekleşen olayları iletmek
- Arayüz ve iletişim modülünden gelen komutları değerlendirmek ve uygun şekilde işlemek

olarak sıralanmaktadır.

4.1.1 Diğer Modüllerin Başlatılması ve Sonlandırılması

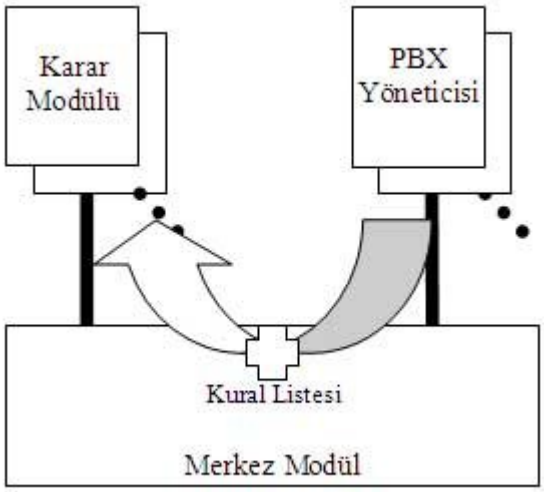
Sistem başlatıldığında ilk olarak merkez modül oluşturulur. Daha sonra arayüz ve iletişim modülleri başlatılır. Bir sonraki adımda, önceden yapılmış bir tanım var ise, veri modülü oluşturulur. Aksi takdirde, kullanıcının kullanılacak veri kaynakları için gerekli tanımları yapması beklenir. Sistemin başlangıcıyla otomatik olarak gerçekleştirilen bu adımlardan sonra PBX modülünü başlatması için komut beklenir. Gerekli komut alnındıktan sonra bağlanılacak PBX tipine uygun olan PBX modülü devreye sokulur. Yönetici PBX ile ilgili tanımlamaları yapmamış ise bunun gerçekleştirilmesi için beklenir. PBX modülü uygun bir şekilde başlatıldıktan sonra bu modülden sisteme bir olay iletilirse, bu olayı karşılayacak bir karar modülü oluşturulur ve ilgili olayla ilişkilendirilir.

Sistem yöneticisi istediği bir anda PBX modülünü durdurabilir. Bu durumda sistem ilk başlangıç durumuna dönmüş olur. Çıkış komutu verilerek sistem kapatılmak istenildiğinde ise sırasıyla karar modülü, PBX modülü, veri modülü, iletişim modülü ve arayüz modülü sonlandırılır ve sistemden çıkış gerçekleştirilir.

4.1.2 PBX Yöneticisinden Gelen Olayların Uygun Bir Karar Modülüne İletilmesi

PBX yöneticisi başlatıldığında sistemdeki kullanıcıların gerçekleştirdiği telefon aktiviteleri ile ilgili bilgileri devamlı olarak sisteme iletmeye başlar. Bu aktivitelerden sistemi ilgilendiren tiptekiler, merkez modül tarafından seçilir ve değerlendirilmek üzere yeni oluşturulan bir karar modülü ile ilişkilendirilir. Karar modülü oluşturulurken sistemde tanımlanmış ilgili kural listesi de Şekil 4.2'de görüldüğü gibi PBX üzerinde gerçekleşen olay ile birlikte bu modüle iletilir.

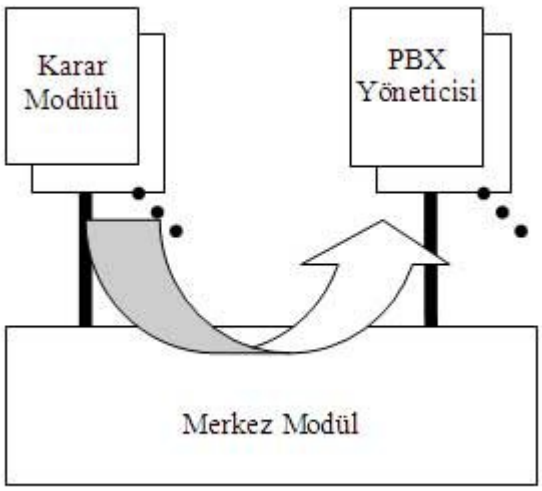
Arka arkaya gelen PBX olaylarını karşılamak üzere birbirleriyle paralel çalışacak karar modülleri devreye sokulur. Eğer sistem kaynaklarının veya tanımlarının getirdiği bir sınırlama var ise gelen olaylar bir kuyruğa sokulur ve yeni bir karar modülünün oluşturulabileceği koşullar oluşuncaya kadar bekletilir.



Şekil 4.2 Olayların karar modülüne iletilmesi

4.1.3 Karar Modülü Tarafından Verilen Komutların PBX Yöneticisine İletilmesi

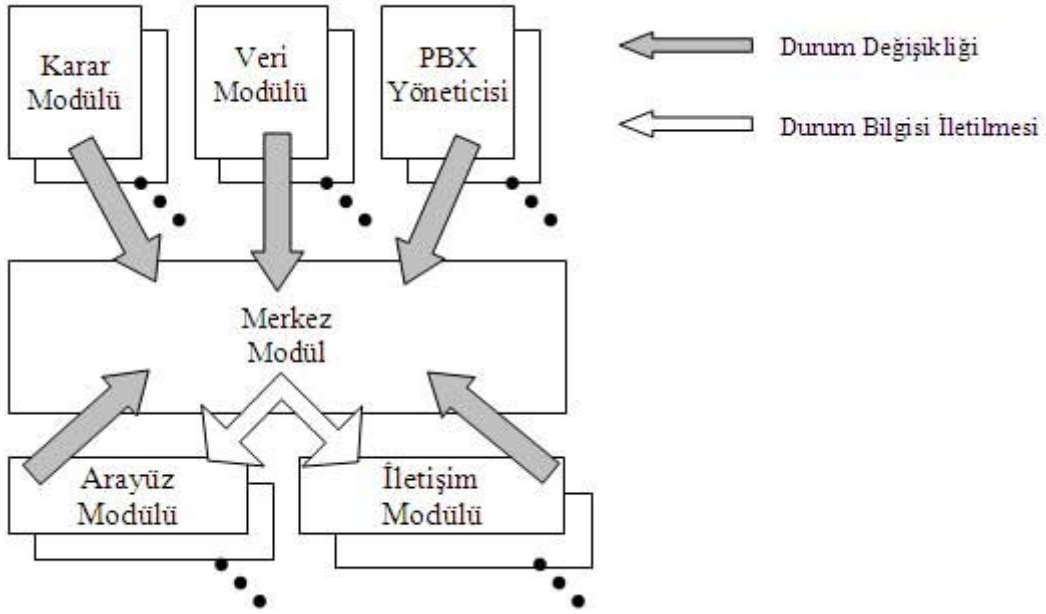
PBX üzerinde gerçekleşen bir olayın karar modülü tarafından değerlendirilmesinin ardından uygulanacak işlem merkez modüle iletilir. Bu işlem merkez modül tarafından değerlendirilir ve Şekil 4.3'te görüldüğü gibi ilgili PBX modülüne iletilir. Sistemde birden fazla PBX yöneticisi mevcut ise merkez modül kararı tetikleyen olayın kaynağı olan PBX yöneticisini belirler ve işlemin ilgili PBX yöneticisine iletilmesini sağlar.



Şekil 4.3 İşlemin PBX modülüne iletilmesi

4.1.4 Arayüz ve İletişim Modüllerine, Sistem Durumunun ve Olayların İletilmesi

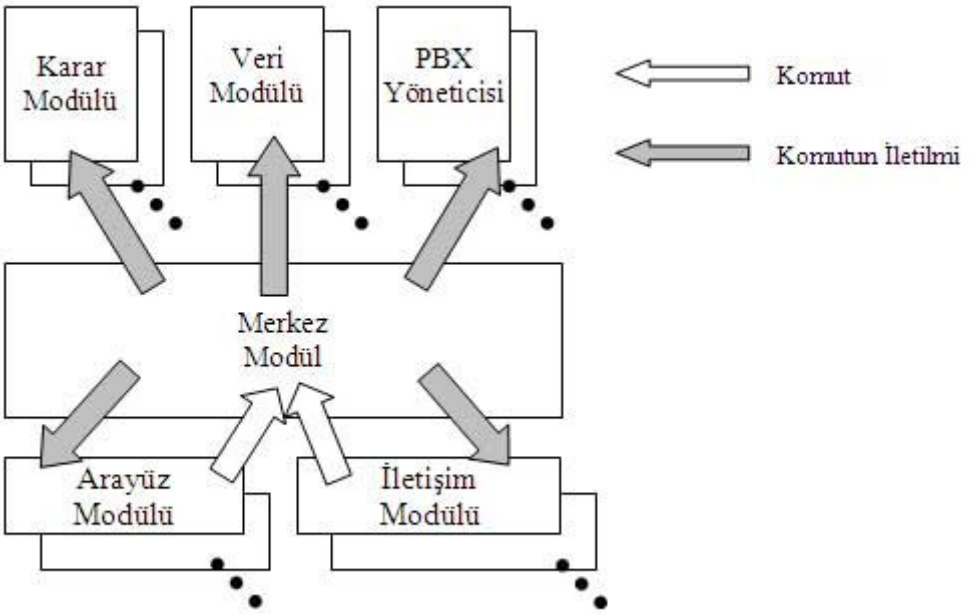
Sistemin çalışması sırasında, sistem dahilinde bulunan modüllerde veya bu modüllerin içinde bulunan yapılarda ortaya çıkan durum değişiklikleri ve olaylar, merkez modül tarafından değerlendirilir. İlgili durum bilgisi, Şekil 4.4'te görüldüğü gibi merkez modül üzerinden arayüz modülüne veya iletişim modülüne iletilir. Örneğin sistemde bir çağrı yapılması, veritabanı bağlantısının kesilmesi veya PBX'in cevap vermemesi gibi bir durum oluştuğunda bu durum yönetici veya kullanıcıya ulaştırılmak üzere ilgili modüle iletilir.



Şekil 4.4 Durum ve olayların iletilmesi

4.1.5 Arayüz ve İletişim Modülünden Gelen Komutların İletilmesi

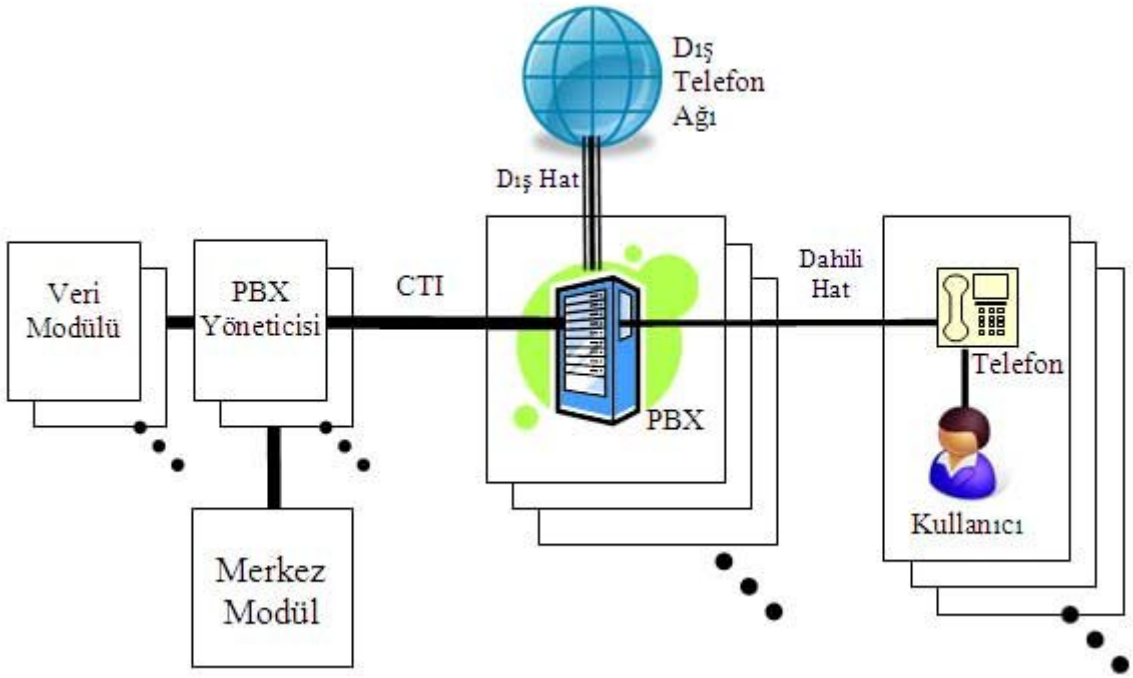
Sistemde gerçekleştirilecek işlemler için verilen komutların arayüz ve iletişim modüllerinden alınması ve ilgili modüllere iletilmesi işi merkez modül tarafından yerine getirilmektedir. Buna göre sistemin çalışması için gerekli tanımların yapılması, modüllerin başlatılması ve sonlandırılması, PBX üzerinden çağrı başlatılması ve sonlandırılması ve çağrının yönlendirilmesi için verilen komutlar, merkez modül tarafından ilgili modüle iletilir ve bu komutların yerine getirilmesi sağlanır.



Şekil 4.5 Komutların iletilmesi

4.2 PBX Yöneticisi

PBX yöneticisi, sistemin PBX ile olan arayüzünü oluşturmaktadır. Farklı altyapı ve protokollere sahip PBX sistemleriyle iletişim kurabilecek ve bu sistemler üzerinde işlem gerçekleştirecek şekilde tasarlanmıştır. Bu sayede diğer modüllerin kullanılan PBX'in tipinden bağımsız olarak çalışabilmesine imkan tanınmaktadır. Şekil 4.5'te PBX modülünün sistemde yer alan diğer yapılarla olan ilişkisi gösterilmektedir.



Şekil 4.6 PBX yöneticisi modülünün bağlantıları

PBX yöneticisi başlangıç anında, bağlantı için gerekli tanımlamaları merkez modülden alır. Bu tanımlamalar çerçevesinde gerekli bağlanıtıyı sağlar ve aktif duruma geçer. Bu aşamada PBX'te gerçekleşen olayları merkez modüle, merkez modülden iletilen komutları da uygun formata çevirerek PBX'e iletmeye başlar. Eğer başlangıç anında PBX ile bağlantı sağlanamazsa veya bağlantı bir süre sonra kesilirse bu durum merkez modüle iletilir ve PBX yöneticisi beklemeye geçer.

Aktif durumdayken PBX yöneticisi, PBX'in marka ve modeline göre değişebilen bir takım olay tiplerini yakalamak üzere PBX'i dinlemeye başlar. Bu olay tipleri,

- Yeni çağrı başlaması (Dış/İç)
- Çağrının sonlanması
- Çağrının yönlenmesi
- PBX'e yeni bir dahili eklenmesi
- Bir dahilinin on-hook durumuna geçmesi
- Bir dahilinin off-hook durumuna geçmesi
- Diğer durum bilgisi değişimi (örn. PBX saatinin değişimi)

olarak sıralanabilir.

PBX modülü bir olayı yakaladığında bunu sistemin değerlendirebileceği bir veri yapısına

dönüştürür. Bu sayede farklı PBX'ler tarafından farklı formatlarda verilen olay bilgisi standart bir yapıda sisteme iletilmiş olur. Bu dönüşüm sırasında genellikle PBX'ler tarafından tanımlanmayan fakat sistemde yapılan ek tanımlar sayesinde belirlenebilen arama tipi, aranan şehir ve ülke ismi gibi bilgilerin de olay bilgisine eklenmesine ek olarak çağrıdaki numaraların standart bloklara ayrılarak işlenmeye hazır hale getirilmesi de sağlanır. Bu dönüşüm sırasında gereken tanımlar doğrudan ilgili veri modülünden alınır ve merkez modüle iletilecek veri yapısı oluşturulur.

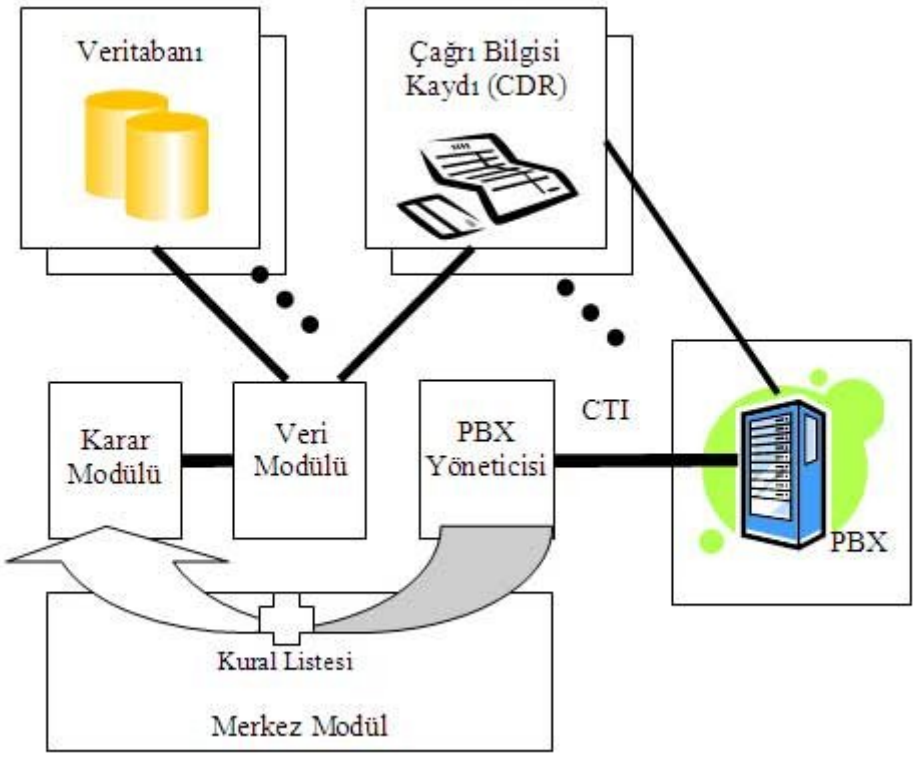
Sistem çapında birden fazla PBX yöneticisi modülü aynı anda görev alabilir. Eğer sistemde birden fazla sayıda PBX var ise her PBX'e uygun tipte bir PBX yöneticisi kopyası sistemin başlangıcı aşamasında merkez modül tarafından oluşturulur ve bağlantıların sağlanması için gerekli tanımlamaların bu modüllere iletilmesi sağlanır.

4.3 Karar Modülü

Karar modülleri, PBX tarafından sisteme iletilen ve sistemin hakkında bir işlem gerçekleştirmesini gerektirebilecek olayların değerlendirilmesini ve yönetilmesini sağlamak üzere birbirine paralel çalışan yapılardır.

Karar modülü, sistemde bir olay oluştuğunda merkez modül tarafından aktif hale getirilir ve ilgili olayın bilgisi ile sistemin kural listesi bu modüle iletilir. Aktif hale gelen karar modülü, olayın mevcut kural listesinin bir elemanına karşılık gelip gelmediğini kontrol eder. Eğer karşılaştırılan kural, sistem tanımlamalarına veya çağrı geçmişine ait bir kontrol gerektiriyorsa, karar modülü veri modülüyle iletişim kurarak gerekli bilgiyi alır. Kontrol sonucu olay ile listenin bir elemanı arasında bir eşleşme oluşuyorsa, kuralın gerektirdiği şekilde bir işlem bilgisi oluşturulur ve merkez modüle iletilir. Bu aşamada merkez modül yeni bir olayın değerlendirilmesi için yeni bir görev verebilir. Aksi takdirde karar modülü sonlandırılır.

Karar mekanizmasın dinamik ve hızlı çalışabilmesi için ardarda gelen PBX olayları için merkez modül tarafından birbirine paralel karar modülleri oluşturulabilir. Sistem kaynaklarının ve tanımlarının sınırladığı sayıda karar modülü, ilgili PBX olaylarıyla ilişkilendirilerek aktif hale getirilir. Daha sonra her karar modülünün döndürdüğü sonuçlar merkez modül tarafından ilgili PBX'e iletilir. Karar modülünün diğer modüllerle olan ilişkisi Şekil 4.7'de gösterilmiştir.



Şekil 4.7 Karar modülünün diğer modüllerle ilişkisi

4.4 Veri Modülü

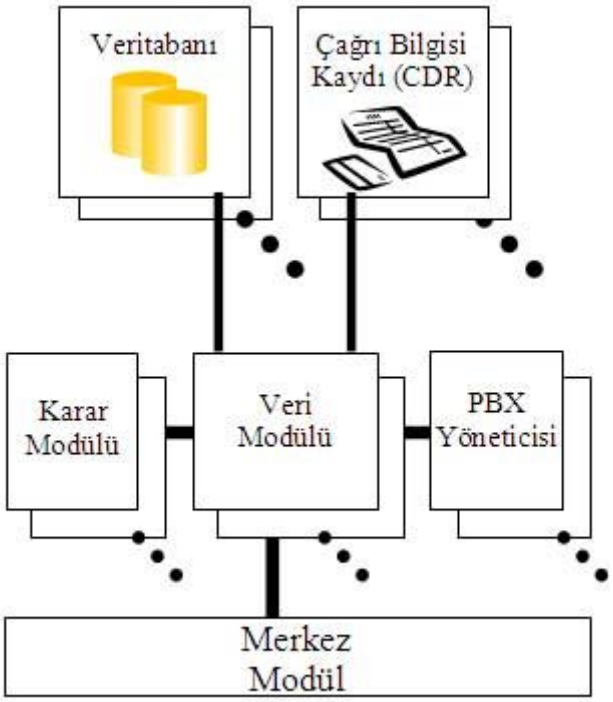
Veri modülü, sistemin ihtiyaç duyduğu çeşitli tipteki bilgilerin farklı kaynaklardan alınması ve sistemin gerektirdiği tanımlama ve konfigürasyon bilgisinin daha sonra kullanılmak üzere saklanması görevini yürüten bir arayüz modülüdür.

Veri modülü sayesinde diğer modüller farklı veri kaynaklarına tek bir arayüzden erişebilme imkanı kazanmaktadır. Bu sayede farklı tipteki veri kaynakları için her modülün kendi implementasyonunu yapması önlenmektedir. Veri modülü, değişen veri kaynaklarından aldığı bilgiyi standart bir veri tipine çevirerek diğer modüllerin kullanımına sunar. Daha sonra ihtiyaç duyan modül bilgiye önceden belirlenmiş metodları kullanarak erişebilir. Veri modülünün sistem içindeki yerleşimi Şekil 4.8'de gösterilmektedir.

Veri modülünün bir diğer işlevi, sistem üzerinde sistem yöneticisi veya kullanıcılar tarafından yapılan tanım ve işlemlerin kayıtlarını, sonradan erişilmek üzere saklamaktır. Bu kayıtlar, önceden belirlenmiş bir veri kaynağı üzerinde belirli veri yapıları kullanılarak kaydedilir ve sistemin gerektirdiği durumlarda tekrar sistemin kullanımına sunulmak üzere geri getirilir.

Tek bir veri modülü farklı veri kaynaklarıyla aynı anda çalışabileceği gibi, sistem

kaynaklarının daha etkin kullanılması ve yüksek performans amacıyla birden çok veri modülünün paralel çalıştırılması da mümkündür. Bu sayede farklı modüllerin aynı veri modülünü kullanması sonucu oluşabilecek darboğaz engellenmektedir.

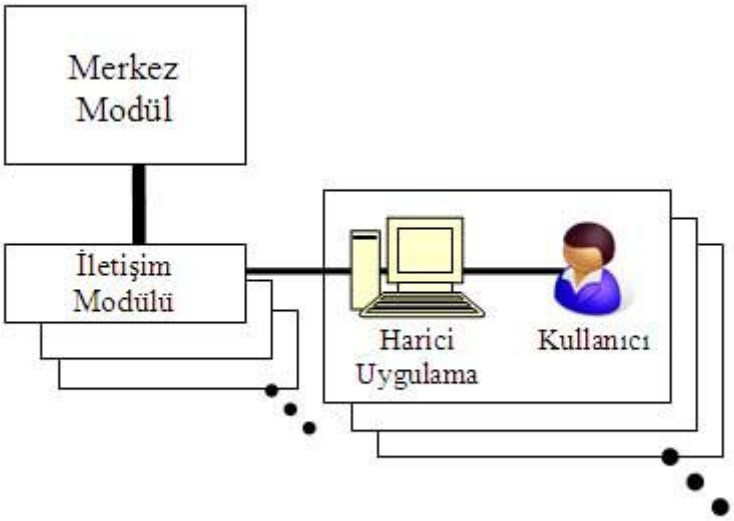


Şekil 4.8 Veri modülünün diğer modüllerle ilişkisi

4.5 İletişim Modülü

İletişim modülü, sistemin sunduğu fonksiyonlara üçüncü şahıs uygulamalar tarafından kontrollü erişimin sağlanmasını sağlayan bir arayüz modülüdür. İletişim modülü sayesinde TCP/IP gibi bir takım protokolleri kullanılarak sesli erişim kontrol sistemi üzerindeki tanımlara, durum bilgisine, olay bilgisine ve komutlara gerçek zamanlı erişim sağlanmakta, bu sayede sistemin kullanım alanı ve çevre sistemler ile olan etkileşimi arttırılmaktadır.

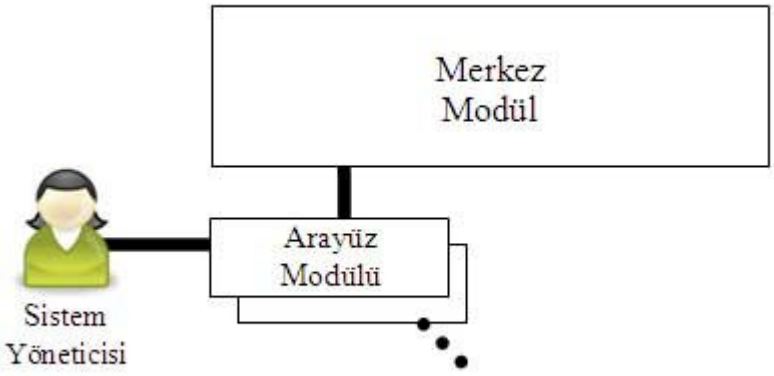
İletişim modülü birden çok sayıda ve farklı tipte bağlantıyı sağlayabilecek yapıdadır. Sistemin farklı fonksiyonlarına erişmek üzere kullanıcılar, farklı uygulamalarla sisteme bağlanmak istediklerinde, bu bağlantıları sağlamak ve yürütmek için birbiriyle paralel çalışabilen iletişim modülleri merkez modül tarafından oluşturulur. Kullanım sonunda bağlantı kesildiğinde iletişim modülü, merkez modül tarafından sonlandırılır. İletişim modülünün sistem ile olan ilişkisi Şekil 4.9'de gösterilmektedir.



Şekil 4.9 İletişim modülünün diğer modüllerle ilişkisi

4.6 Arayüz Modülü

Arayüz modülü, sistem üzerindeki bilgi, tanımlama ve fonksiyonlara ara bir bağlantı kullanmadan doğrudan erişim sağlamaktadır. Arayüz fonksiyonları ve tasarımı, sistemde ayrı bir modül olarak tasarlanarak implementasyonda karşılaşılabilecek bağlılık en alt düzeye indirilmiş, arayüzde soyutlama sağlanmıştır. Bu özellik sisteme gerektiğinde farklı arayüzlerden erişilebilmesineimkanı sağlamaktadır (Şekil 4.10).



Şekil 4.10 Arayüz modülünün diğer modüllerle ilişkisi

Yönetim arayüzü, sistem tanımlarının yapılmasını, sistemin başlatılmasını ve

sonlandırılmasını, sistem üzerinde gerçekleşen olayların izlenmesini ve yönetilmesini sağlayacak kontrolleri içermektedir. Bu kontroller ilgili fonksiyonlara göre organize edilmiş ve kullanım kolaylığı esas alınmıştır. Arayüz modülü, ileride ihtiyaç duyulabilecek yeni fonksiyon ve kullanım tiplerine göre kolay ve hızlı bir şekilde dönüştürülebilir ve geliştirilebilir yapıda tasarlanmıştır. Sistem yönetisinin kullanım alışkanlıklarına göre özelleştirilebilmesi mümkün olmaktadır.

5. UYGULAMA

Sisteme ait uygulama, tasarımda ortaya konulan sesli iletişim kontrolü modül ve fonksiyonlarının gerçekleştirilmesi sonucu ortaya çıkmıştır. Kullanılan araçlar ve geliştirilen modüller ile ilgili ayrıntılı bilgi aşağıda verilmektedir. Uygulamaya ait sınıf diyagramları ve veri yapılarına ait ayrıntılı bilgi altıncı bölümde yer almaktadır.

5.1 Gerçeklenen Sistemde Kullanılan Araçlar

Tasarlanan sistemin gerçeklenmesi aşamasında kullanılan araçlar, tez çalışmasının birinci ve ikinci bölümünde açıklanan ihtiyaç ve amaçlar doğrultusunda seçilmiştir. Uygulama geliştirme dili olarak JAVA, PBX sistemi olarak Asterisk seçilmiştir. CTI API'si olarak Asterisk-JAVA kütüphanesi kullanılmıştır. Sistemin veri kaynağı olarak Microsoft SQL Server Desktop Engine tercih edilmiştir. Kullanılan bu araçlar ve seçim kriterleri hakkında bilgi aşağıda verilmiştir.

5.1.1 JAVA

Sistemin uygulamasında, geliştirme dili olarak JAVA tercih edilmiştir. Yapılan tercihte JAVA'yı ön plana çıkaran özellikler şu şekilde sıralanabilir:

- Platformdan bağımsız çalışabilmesi: JAVA ile geliştirilen uygulamalar, farklı platformlar üzerinde çalıştırılabilmektedir.
- Nesneye yönelik olması: JAVA, hiyerarşik yapıda tasarlanan sistemlerin, veri yapılarının ve bu yapılar arası ilişkilerin gerçeklenmesi için uygun bir geliştirme yöntemi sunmaktadır.
- Sağlamlık: İstisna durumların (Exceptions) ve veri tiplerininin yanı sıra başta hafıza olmak üzere tüm sistem kaynaklarına erişimin sıkı bir şekilde kontrol edilmesi geliştirilen uygulamaların sağlamlığına katkıda bulunmaktadır.
- Thread ve Ağ desteği: JAVA birden çok sayıda thread kullanan uygulamaların geliştirilmesine imkan tanıyan güçlü fonksiyonlar sağlamaktadır. JAVA, tasarımı itibarıyla kapsamlı ağ uygulamalarının geliştirilmesine yatkındır.

Yukarıda verilen temel özellikler, farklı sistemlerle uyum içinde çalışması esas alınan ve güçlü, hiyerarşik yapıda bir ağ uygulaması olan sesli iletişim kontrolü uygulamasının gerçeklenmesinde geliştirme dili olarak JAVA'nın seçilmesini sağlamıştır.

5.1.2 Asterisk PBX

Gerçeklenen sistemin PBX bileşeni olarak Asterisk PBX seçilmiştir. Asterisk'in tercih edilmesinde etkili olan başlıca özellikler şu şekilde sıralanabilir:

- Hem geleneksel hem de VoIP telefon ağlarıyla birlikte çalışabilmesi: Bu özelliği sayesinde Asterisk farklı telefon altyapılarına sahip kurumlarda kullanılabilmektedir. Bunu sağlamak adına Asterisk dahilinde geniş bir protokol ve codec desteği bulunmaktadır.
- Geniş fonksiyonalite: Asterisk, genellikle ileri seviye ve yüksek maliyetli telefon sistemleriyle ilişkilendirilen telefon hizmetlerinin verilebilmesine imkan tanımaktadır. Böylece geniş bir kullanım alanı ortaya koyulmaktadır.
- Ölçeklenebilirlik: Konfigürasyonundaki esneklik sayesinde Asterisk, farklı ölçekteki kurumlarda kullanılabilmektedir.
- Gelişmiş CTI özellikleri: Desteklediği CTI arayüzleri sayesinde Asterisk'le ortak çalışan uygulamaların geliştirilmesine imkan tanınmaktadır.

Sesli haberleşme sistemi kontrolü uygulaması, farklı telefon altyapılarıyla çalışabilen, gelişmiş, ölçeklenebilir bir CTI uygulamasıdır. Bu nedenle gerçeklenen uygulamada yukarıdaki özellikleriyle ön plana çıkan Asterisk PBX kullanılmıştır.

5.1.3 Asterisk-JAVA API

Asterisk-JAVA API, Asterisk PBX sistemiyle çalışan bir JAVA uygulaması geliştirmek için gereken alt seviye CTI iletişim fonksiyonlarını içeren sınıflar paketidir. API dahilindeki fonksiyonlar, bağlantının kurulması ve mesajlaşma gibi Asterisk CTI protokolüne has özel işlemleri yerine getirmektedir. Asterisk-JAVA, aynı zamanda farklı versiyonlardaki Asterisk PBX'lere standart bir arayüz üzerinden erişebilmeye imkan tanımaktadır. Bu da kurumdaki Asterisk PBX'in üzerinde gerçekeştirilebilecek değişikliklerden uygulamanın etkilenmesini engellemektedir.

5.1.4 Microsoft SQL Server Desktop Engine

Tez çalışması sonucunda gerçeklenen uygulamada veri kaynağı olarak Microsoft SQL Server Desktop Engine (MSDE) kullanılmıştır. Seçimde etkili olan başlıca özellikler şunlardır:

- Ücretsiz kullanılabilmesi: MSDE, ek bir lisanslamaya gerek duyulmadan ücretsiz olarak dağıtılabilmekte ve kullanılabilmektedir.
- İlişkisel veritabanı olması: MSDE, ilişkisel veritabanı yapısına sahiptir. Buna göre tablolar arası ilişkilerin veritabanı üzerinde tanımlanması gibi uygulamanın veri yapısına katkıda bulunabilecek özellikleri gerçeklemek MSDE ile mümkün olmaktadır.
- Farklı CDR uygulamalarıyla uyumluluk: Telefon sistemlerinde kullanılan CDR uygulamalarının MSDE ile uyumlu çalışması, gerçeklenen sisstemin CDR sistemleriyle entegrasyonuna katkıda bulunmaktadır.

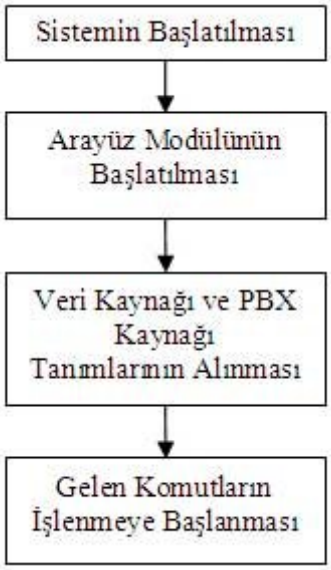
Yukarıda değinilen özellikler kapsamında, gerçeklenen uygulamanın temel veri kaynağı olarak MSDE seçilmiştir.

5.2 Gerçeklenen Modüller

Tez çalışmasının tasarım aşamasında ortaya konulan sistemin belirlenen senaryolar kapsamında gerçeklenmesi sonucu oluşturulan modüller bu bölümde anlatılmaktadır.

5.2.1 Merkez Modül

Merkez modül, sistemin diğer modüllerini başlatır ve birbirleriyle olan iletişimlerini sağlar. Merkez modülün başlatılması sonucu öncelikle arayüz modülü oluşturulur ve başlatılır. Arayüz modülünün oluşturulmasının hemen ardından sistem tanımlarından veri kaynağı ve PBX kaynağı için gerekli bağlantı parametreleri alınır. Daha sonra merkez modül, arayüz modülünden komut beklemeye başlar. Sistemin başlangıç adımları şekil 5.1'de gösterilmektedir.



Şekil 5.1 Merkez modülün başlangıç adımları

Merkez modül başlatıldıktan sonra bekleme durumuna geçer. Bu aşamada sistem yöneticisi arayüz modülünden merkez modüle komut gönderebilir. Merkez modülüne gönderilebilecek komutlar şu şekilde sıralanabilir:

- **connectPBXManager:** PBX'e bağlantı kurulmasını sağlar
- **startPBXManager:** PBX'ten olayların alınmasını başlatır.
- **stopPBXManager:** PBX olaylarının alınmasını durdurur.
- **disconnectPBXManager:** PBX bağlantısının kesilmesini sağlar

Sistem yöneticisi öncelikle PBX'e bağlantı kurulması için bir komut yollar. Eğer önceden

kurulmuş bir bağlantı yoksa, bağlantı yapılması için PBX yöneticisi modülüne komut gönderilir. Eğer önceden oluşturulmuş bir bağlantı varsa, komut dikkate alınmaz. Bağlantı kurulması sırasında bir hata oluşursa sistem kayıtlarına hatayla ilgili açıklama kaydedilir ve sistem yöneticisine bildirilir.

PBX bağlantısının kurulmasının hemen ardından bir veri modülü oluşturulur. PBX yöneticisinin ihtiyaç duyacağı bilgileri doğrudan alması için oluşturulan veri modülü PBX yöneticisiyle ilişkilendirilir.

PBX bağlantısının kurulmasından sonra sistem yöneticisi, PBX olaylarının işlenmeye başlaması için komut yollayabilir. Bu komut sonrası merkez modül PBX yöneticisinden gelecek olay paketlerini beklemeye başlar. Tanımlı bir olay paketinin gelmesi sonucu merkez modül bir karar modülü oluşturur ve gelen olay paketini mevcut kural listesiyle birlikte oluşturulan karar modülüne iletir.

İstenilen bir zamanda sistem yöneticisi olayların dinlenmesini durdurabilir. Merkez modülü ilgili komutu PBX yöneticisine aktarır ve olayların merkez modülüne iletilmesi sonlandırılır. PBX yöneticisi bu noktada PBX ile olan bağlantıyı sürdürmeye devam eder fakat PBX'ten iletilen olay bilgisi merkez modüle iletilmez.

Sistem yöneticisi, merkez modüle bir komut göndererek PBX bağlantısını sonlandırabilir. Bu komut merkez modül tarafından PBX yöneticisine iletilir ve PBX bağlantısı kesilir. Bağlantı kesildikten sonra yeni bir bağlantı kurulana kadar PBX yöneticisi üzerinden PBX'e yeni bir komut gönderilemez.

Merkez modül, PBX yöneticisiyle tanımlı bir arayüz üzerinden iletişim kurmaktadır. Uygulamada kullanılan Asterisk PBX ile olan iletişim tamamen AsteriskPBXManager sınıfı tarafından gerçekleştirilir. Bunu sağlamak için merkez modül, AsteriskPBXManager sınıfının implemente ettiği PBXInterface arayüzü ile çalışmaktadır. Merkez modülün kullandığı sınıf ve nesneler ve yapı diyagramları tez çalışmasının altıncı bölümde verilmiştir.

5.2.2 Asterisk PBX Yöneticisi

Asterisk PBX yöneticisi, Asterisk PBX ile iletişim kurmak için özelleştirilmiş bir PBX yöneticisi modülüdür. Diğer modüller, Asterisk PBX yöneticisiyle, standart PBX yöneticisi arayüzüyle iletişim kurarlar. Böylece, Asterisk PBX'e özel işlemler bu modül tarafından diğer modüllerden soyutlanarak gerçekleştirilir.

Asterisk PBX Yöneticisi, PBX yöneticisi modülünün sunduğu tüm komutları içermektedir. Bu komutlar şu şekilde sıralanabilir:

- **connect:** PBX ile bağlantı kurulmasını sağlar.
- **disconnect:** PBX ile olan bağlantıyı sonlandırır.
- **run:** PBX'ten gelen olayların kayıtlı dinleyicilere iletilmesini başlatır.
- **stop:** PBX'ten gelen olayların kayıtlı dinleyicilere iletilmesini sonlandırır.
- **hangup:** Verilen hatlar üzerindeki görüşmeyi sonlandırır.
- **redirect:** Verilen hattı başka bir numaraya bağlar.
- **isConnected:** PBX ile olan bağlantı durumunu gösterir.
- **isRunning:** PBX olaylarının iletilme durumunu gösterir
- **addEventListener:** PBX olaylarının iletileceği yeni bir dinleyici kaydedilir.
- **removeEventListener:** PBX olaylarının iletildiği bir dinleyici devre dışı bırakılır.

PBX ile bağlantı kurulduktan sonra çağrı başlatma, yönlendirme veya sonlandırma gibi komutlar bu bağlantı üzerinden PBX'e gönderilebilir. Buna ek olarak PBX üzerindeki olaylar Asterisk PBX yöneticisi tarafından alınmaya başlar. Fakat olay iletimi başlatılmadıysa gelen olaylar merkez modüle gönderilmez ve yok sayılır.

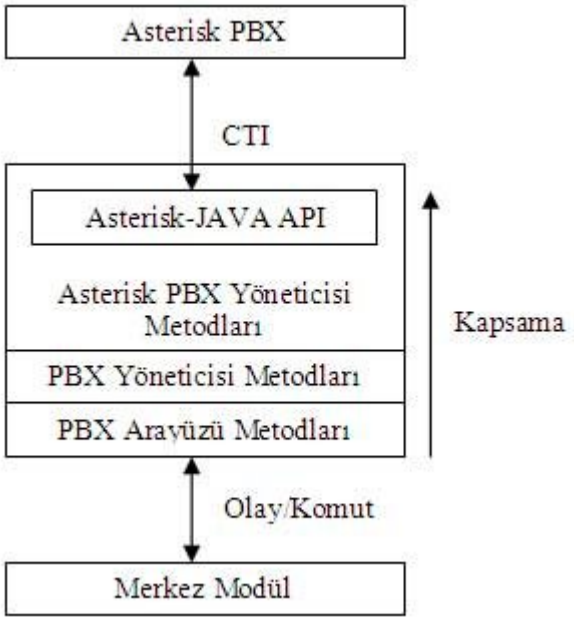
Olay iletimi başlatıldıktan sonra bir çağrı olayı geldiğinde, Asterisk PBX yöneticisi öncelikle sistem tanımlarından aldığı context bilgisiyle PBX üzerinde başlatılan çağrıların tanımlaması işlemini gerçekleştirir. Veri kaynağından da yararlanarak başlatılan çağrının yön, tip ve diğer özellikleri tanımlandıktan sonra standart bir çağrı olayı yapısı oluşturularak merkez modüle iletilir.

PBX yöneticisinin içerdiği dinleyici mekanizması, geliştirilebilecek başka modüllerin de gerektiğinde PBX yöneticisinin ilettiği olayları doğrudan almasına imkan tanımaktadır. Bunu sağlamak için *addEventListener* metodunun ilgili modül tarafından gerektiği şekilde kullanılması yeterlidir. Böylece istenilen bir komut seti, iletilen olayla birlikte çalıştırılabilmektedir.

Merkez modül istenilen bir anda olay iletimini durdurabilir. Yine aynı şekilde kurulan bağlantıyı kesebilir. Bağlantı kesildikten sonra Asterisk PBX yöneticisi sonlandırılmaz. Merkez modül tarafından kullanılmak üzere bekletilir. Merkez modül daha sonra bu yönetici üzerinden yeni bir bağlantı kurabilir veya yöneticiyi sonlandırarak sistem kaynaklarını geri kazandırabilir.

Asterisk PBX yöneticisi, Asterisk PBX ile olan CTI bağlantısını Asterisk-JAVA API'si üzerinden sağlamaktadır. Bu API, Asterisk'in Manager olarak adlandırılan arabirimiyle Telnet bağlantısı kurar ve alt seviye komut iletimini gerçekleştirir. Asterisk-JAVA, Manager'in farklı

versiyonlarıyla iletişim kurabilmektedir. Bu da sistemin farklı Asterisk versiyonlarıyla olan uyumluluğuna katkıda bulunmaktadır. Asterisk PBX yöneticisinin katmanları Şekil 5.2'de gösterilmektedir.



Şekil 5.2 Asterisk PBX Yöneticisi katmanları

Asterisk PBX yöneticisi ile ilgili sınıf ve metodlar altıncı bölümde ayrıntılı olarak verilmektedir.

5.2.3 Microsoft SQL Server Veri Yöneticisi

Microsoft SQL Server Veri Yöneticisi, Microsoft SQL Server ile iletişim kurmak üzere özelleşmiş bir veri yöneticisi modülüdür. Diğer modüller, Microsoft SQL Server veri yöneticisiyle, standart veri yöneticisi arayüzü üzerinden iletişim kurarlar. Böylece, Microsoft SQL Server'e özel sorgular bu modül tarafından diğer modüllerden soyutlanarak çalıştırılır.

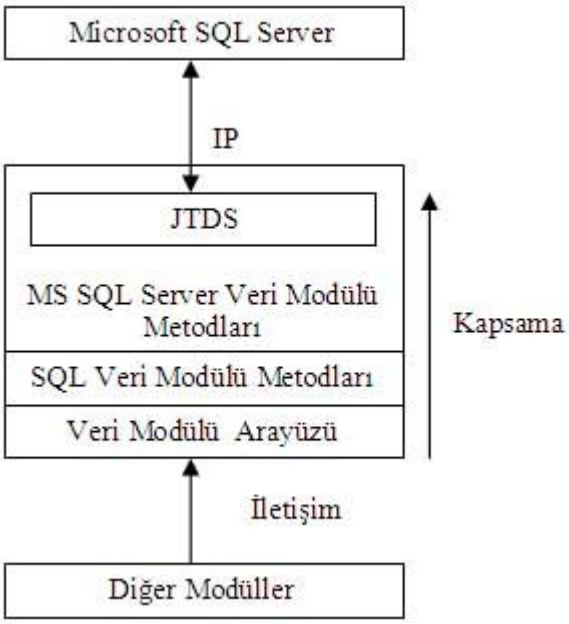
Microsoft SQL Server veri yöneticisi, Veri yöneticisi modülünün sunduğu tüm metodları içermektedir. Bu metodlar şu şekilde sıralanabilir:

- **openConnection:** SQL Server ile üzerinden sorgu yapılabilecek bir bağlantı kurulmasını sağlar.
- **closeConnection:** SQL Server ile kurulmuş bağlantıyı sonlandırır.
- **testConnection:** SQL Server bağlantısının durumunu kontrol eder.
- **getRules:** Sesli iletişim kontrolü sistemi için tanımlı kural listesini veri kaynağından alır.
- **setRules:** Kullanılan kural listesini daha sonra tekrar kullanmak üzere veri kaynağına kaydeder.
- **getNumberType:** Bir telefon numarasının tipini sistem rehberinden alır.

- **getPorts:** Sistemde tanımlı portların listesini veri kaynağından alır.

Sistemin başlatılmasıyla beraber merkez modül sistem tanımlarından yararlanarak mevcut veri kaynağıyla uyumlu bir veri modülünü başlatmak üzere gerekli işlemleri yerine getirir. Sesli iletişim kontrolü uygulaması modüllerinden biri veri kaynağına erişmek istediğinde merkez modül, veri modülünü, standart veri modülü arayüzüyle başlatır ve talepte bulunan modüle bu arayüz aktarılır. Böylece modüllerin sistemdeki veri kaynağının tipinden bağımsız olarak işlem gerçekleştirmesi sağlanır.

Uygulamada, veri modülüyle Microsoft SQL Server veritabanı arasındaki iletişim JAVA Tabular Data Streams (JTDS) [24] sürücüsü ile sağlanmaktadır. JTDS, SQL Server ile hızlı ve etkin iletişim sağlamakta, sistem performansına katkıda bulunmaktadır. Microsoft SQL Server veri modülünün uygulama katmanları Şekil 5.3'te gösterilmektedir.



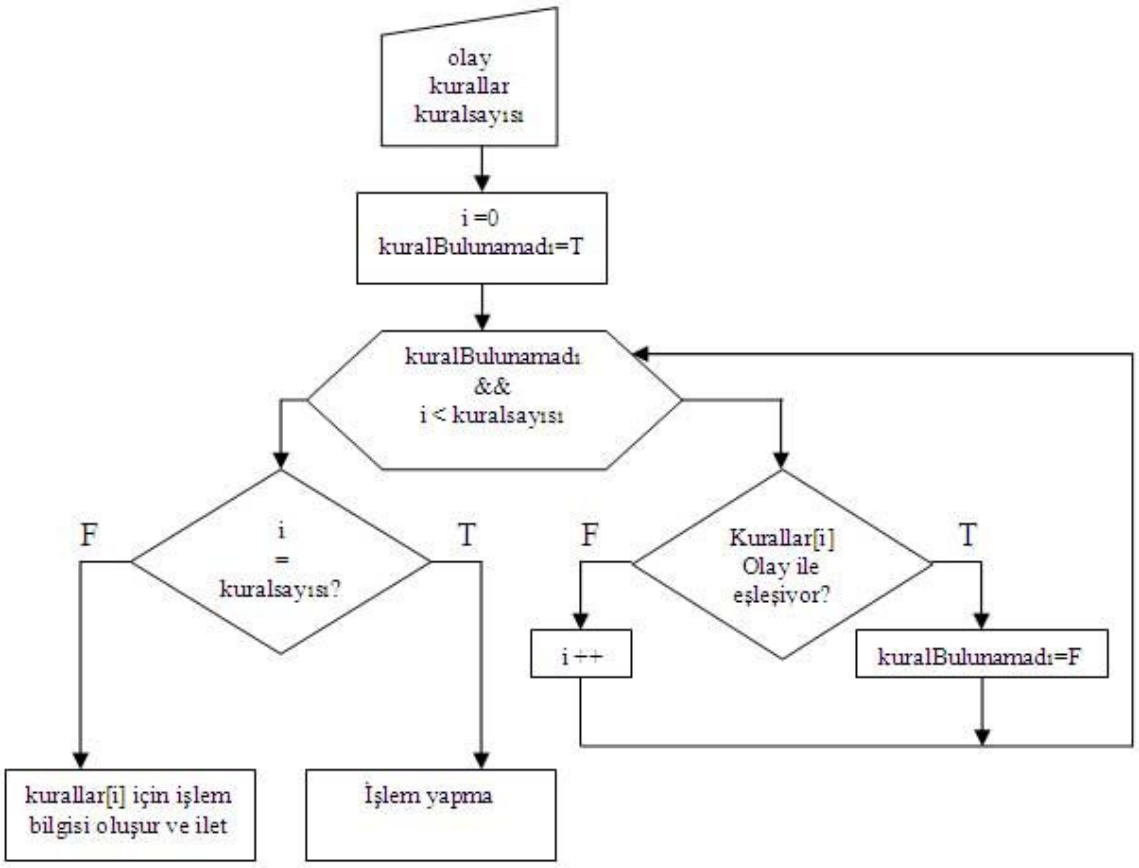
Şekil 5.3 Microsoft SQL Server veri modülü katmanları

Microsoft SQL Server veri modülüyle ilgili sınıf, metot ve diyagramlar altıncı bölümde ayrıntılı olarak verilmektedir.

5.2.4 Karar Modülü

Karar modülü, merkez modülün ilettiği kontrol sistemi açısından anlamlı PBX olaylarının değerlendirildiği ve yapılacak işlemin belirlendiği bileşendir.

Merkez modül anlamlı her PBX olayı için bir karar modülü oluşturur ve devreye sokar. Oluşturulan karar modülleri, birbirlerinden bağımsız olarak ilişkilendirildikleri olayları kendilerine aktarılan kural listesiyle karşılaştırır ve yerine getirilecek işlemi belirler. Eğer kural listesinde iletilen olayla eşleşen bir tanım yoksa modül işlem yapmadan sonlandırılır (Şekil 5.4).



Şekil 5.4 Karar modülü akış diyagramı

Gerçeklenen sistemde karar işlemleri çağrının yönlendirilmesi ya da sonlandırılması şeklinde belirlenmiştir. Sistemde kullanılan PBX'in yetenekleriyle paralel olarak farklı sistemler için yeni işlemler de eklemek mümkündür. PBX'e erişim merkez modül üzerinden iletilen standart PBX yöneticisi arayüzü ile olduğundan işlemlerin yerine getirilmesi için karar modülünde ek bir tanıma ihtiyaç duyulmamaktadır.

Karar modülleri kendi thread'inde çalışacak şekilde tasarlanmıştır. Merkez modül sistem kaynakları dahilinde ihtiyaç duyulan sayıda karar modülünü devreye sokabilir. Görevini yerine getiren karar modülleri kendiliğinden sonlandırılır ve kullandıkları kaynaklar sisteme iade edilir.

Karar modülüyle ilgili ayrıntılı sınıf ve metot diyagramları tez çalışmasının altıncı bölümünde verilmektedir.

5.2.5 Arayüz Modülü

Arayüz modülü, sistem yöneticisinin sistemi başlatma ve sonlandırma gibi komutları verebildiği ve verilen komutların sonuçlarını izleyebildiği bileşendir. Arayüz modülü, farklı tipteki kullanıcı girişlerini destekleyebilecek ve geliştirilebilir yapıda gerçeklenmiştir.

Arayüz modülü dahilinde gerekli bileşenlerin oluşturulması için JAVA Standard Widget Toolkit kütüphanesinden yararlanılmıştır. Bu sayede platformdan bağımsız ve hızlı geliştirilebilir bir arayüz gerçeklenmiştir.

Arayüz modülünün ekran görüntüsü Şekil 5.5'te verilmektedir.

Şekil 5.5 Uygulama arayüzü

Geliştirilen uygulamada arayüz modülü altı fonksiyonu desteklemektedir. Bunlar şu şekilde sıralanabilir:

- PBX bağlantısının başlatılması
- PBX bağlantısının sonlandırılması
- Olay dinleme mekanizmasının başlatılması
- Olay dinleme mekanizmasının sonlandırılması
- Kural listesinin güncellenmesi
- Uygulamadan çıkış

Arayüz modülüyle ilgili ayrıntılı sınıf ve metot diyagramları tez çalışmasının altıncı bölümünde verilmiştir.

6. PAKET VE SINIFLAR

Sesli haberleşme sistemleri için erişim kontrolü uygulaması yedi JAVA paketinden oluşmaktadır. Bu paketler şu şekilde sıralanabilir:

- Ana paket (tfirewall)
- Beans paketi (tfirewall.beans)
- SQL paketi (tfirewall.sql)
- Judgement paketi (tfirewall.judgement)
- PBX paketi (tfirewall.pbx)
- Events paketi (tfirewall.pbx.events)
- Utility paketi (tfirewall.util)

Her pakette ilgili sınıflar yer almakta ve uygulama kodunun organizasyonu sağlanmaktadır.

6.1 Ana Paket

Bu pakette uygulamanın başlatılmasını sağlayan MainWindow sınfını yer almaktadır. Uygulamanın merkezi fonksiyonlarını oluşturan yapılar bu paket içinde yer almaktadır.

6.1.1 MainWindow Sınıfı

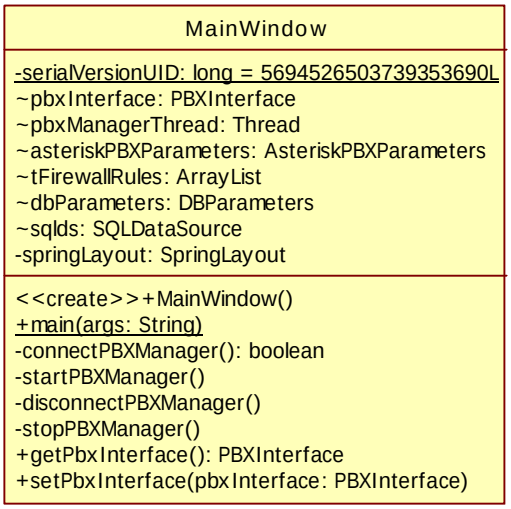
MainWindow sınıfı uygulamanın başlatıldığı, merkez ve arayüz modülü fonksiyonlarının gerçeklendiği sınıftır.

MainWindow sınıfı 8 metot içermektedir (Şekil 6.1). Bu metodlar şu şekilde sıralanabilir:

- **MainWindow metodu:** Sınıfın constructor metodudur. Uygulama başlatıldığında yapılacak işlemleri yerine getirir.
- **main metodu:** Uygulamanın başlatıldığında çalışan metoddur. Temel sistem çağrılarını yerine getirdikten sonra MainWindow() metodunu çalıştırır. Bu metot String tipinde başlangıç parametresi alabilmektedir.
- **connectPBXManager metodu:** PBX yöneticisinin bağlantı kurması için gerekli komutları bu modüle gönderir.
- **startPBXManager metodu:** PBX yöneticisinin PBX olaylarını iletmesi işlemini başlatacak komutları bu modüle iletir.
- **disconnectPBXManager metodu:** PBX yöneticisinin bağlantısının kesilmesi için gerekli komutları bu modüle iletir.
- **stopPBXManager metodu:** PBX yöneticisinin PBX olaylarını iletmesini durduracak komutları bu modüle iletir.
- **getPBXInterface metodu:** PBX yöneticisi arayüzüne erişim sağlar.
- **setPBXInterface metodu:** Aktif PBX yöneticisi arayüzünü tanımlar. Parametre olarak kullanılacak PBXInterface tipindeki arayüz objesini kabul eder.

MainWindow sınıfı 8 adet özellik içermektedir (Şekil 6.1). Bu özellikler şu şekilde sıralanmaktadır:

- **serialVersionUID:** long tipindeki bu özellik sınıfın serialize edilmesi işleminde kullanılır.
- **pbxInterface:** PBXInterface tipindeki bu özellik sistemin PBX fonksiyonlarına erişimi için kullanılan arayüz objesine referans vcrir.
- **pbxManagerThread:** PBX modülünün çalıştığı thread'e referans verir. Bu thread üzerinde yapılan işlemlerde kullanılır.
- **asteriskPBXParameters:** Asterisk PBX'e bağlantı kurulması için gereken tanımları tutar.
- **tFirewallRules:** ArrayList tipindeki bu özellik karar mekanizması için kullanılan kural listesini tutar.
- **dbParameters:** Veritabanı bağlantısının kurulması için gerekli tanımları tutar.
- **sqlds:** SQL Veritabanı üzerinde gerçekleştirilebilecek fonksiyonlara erişimin sağlandığı arayüz sınıfına referans verir.
- **springLayout:** Kullanıcı arayüzü ile ilgili yapıları tutar.



Şekil 6.1 MainWindow sınıfı

MainWindow sınıfı uygulama başlatıldığında ilk oluşturulan sınıftır ve uygulama sonlandırılana kadar ayrı bir thread'de görev alır.

6.2 Beans Paketi

Beans paketi, uygulamada kullanılan veri yapılarının saklanması ve iletilmesi için gerekli sınıfları içermektedir. Beans paketi yedi sınıftan oluşmaktadır (Şekil 6.2). Bu sınıflar şu şekilde sıralanmaktadır:

- PBXParameters sınıfı
- AsteriskPBXParameters sınıfı

- DBParameters sınıfı
- PBXCall sınıfı
- Port sınıfı
- Rule sınıfı
- TelNumber sınıfı

6.2.1 PBXParameters Sınıfı

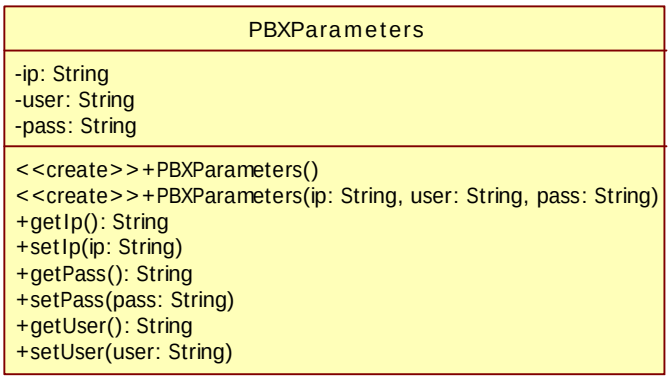
PBXParameters, PBX bağlantısı için gerekli temel tanımların tutulduğu sınıftır.

PBXParameters sınıfı sekiz adet metot içermektedir (Şekil 6.2). Bu metodlar şu şekilde sıralanmaktadır:

- **PBXParameters:** Sınıfın constructor metodudur. Sınıfın null özelliklerden oluşan boş bir kopyasını oluşturur.
- **PBXParameters:** Sınıfın ikinci constructor metodudur. Sınıfın verilen değerlere sahip özelliklerinden oluşan bir kopyasını oluşturur.
- **getIp:** PBX'e bağlantı için kullanılan IP adresini verir.
- **setIP:** PBX'e bağlantı için kullanılan IP adresini kaydeder.
- **getPass:** PBX'e bağlantı için kullanılan şifreyi verir.
- **setPass:** PBX'e bağlantı için kullanılan şifreyi kaydeder.
- **getUser:** PBX'e bağlantı için kullanılan kullanıcı adını verir.
- **setUser:** PBX'e bağlantı için kullanılan kullanıcı adını kayededer.

PBXParameters sınıfı 3 özellik içermektedir (Şekil 6.2). Bu özellikler şu şekilde sıralanmaktadır:

- **ip:** PBX'e bağlantı için kullanılan IP adresini tutar.
- **user:** PBX'e bağlantı için kullanılan kullanıcı adını tutar.
- **pass:** PBX'e bağlantı için kullanılan şifreyi tutar.



Şekil 6.2 PBXParameters sınıfı

PBXParameters sınıfı, farklı PBX tipleri için bağlantı tanımlarını tutacak sınıflar tarafından genişletilir.

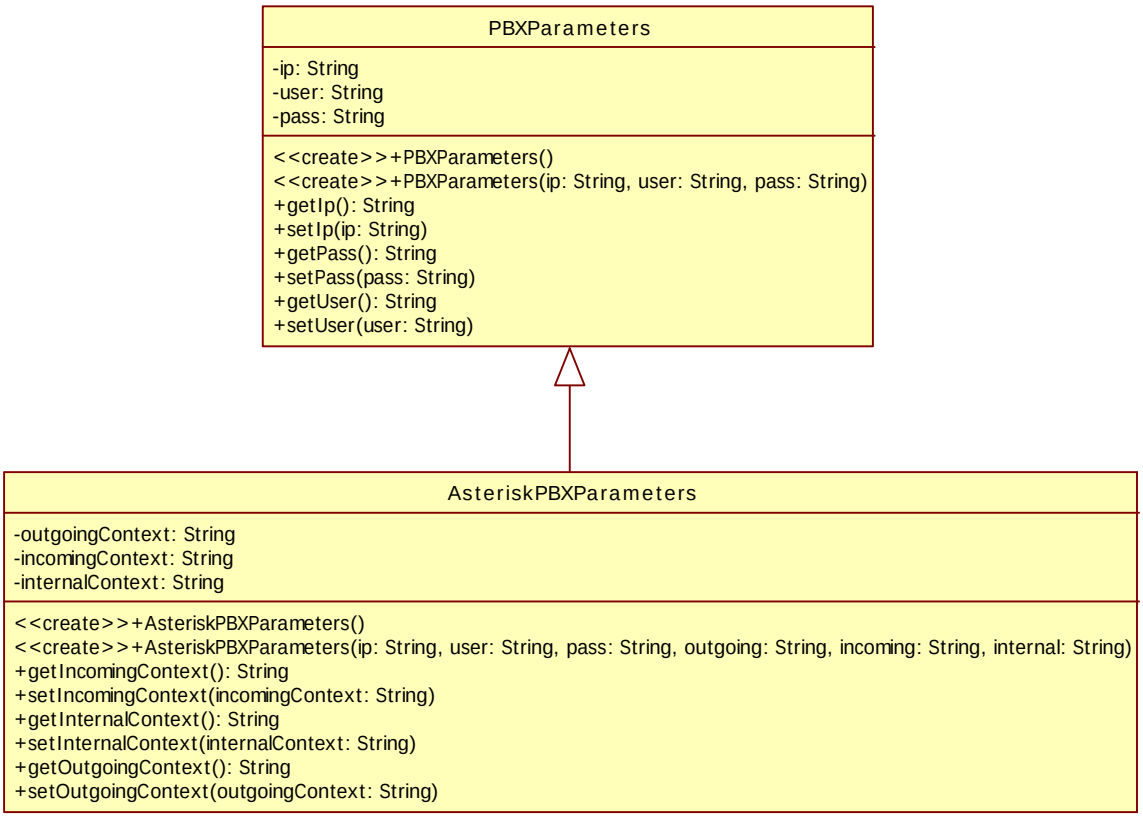
6.2.2 AsteriskPBXParameters Sınıfı

AsteriskPBXParameters, Asterisk PBX'e bağlanmak için gerekli tanımları tutan sınıftır. PBXParameters sınıfına 8 metodun eklenerek genişletilmesi ile oluşturulmuştur (Şekil 6.3). Bu metodlar şu şekilde sıralanmaktadır:

- **AsteriskPBXParameters:** Sınıfın construtor metodudur. Sınıfın null değerlere sahip özelliklerden oluşan boş bir kopyasını oluşturur.
- **AsteriskPBXParameters:** Sınıfın ikinci constructor metodudur. Sınıfın verilen değerlere sahip özelliklerden oluşan bir kopyasını oluşturur.
- **getIncomingContext:** PBX'in gelen çağrılar için tanımlanmış çağrı planı ismini verir.
- **setIncomingContext:** PBX'in gelen çağrılar için tanımlanmış çağrı planı ismini kaydeder.
- **getInternalContext:** PBX'in dahili çağrılar için tanımlanmış çağrı planı ismini verir.
- **setInternalContext:** PBX'in dahili çağrılar için tanımlanmış çağrı planı ismini kaydeder.
- **getOutgoingContext:** PBX'in giden çağrılar için tanımlanmış çağrı planı ismini verir.
- **setOutgoingContext:** PBX'in giden çağrılar için tanımlanmış çağrı planı ismini kayeder.

AsteriskPBXParameters sınıfında, PBXParameters sınıfına ek olmak üzere 3 özellik bulunmaktadır. Bu özellikler şunlardır:

- **outgoingContext:** PBX'in giden çağrılar için tanımlanmış çağrı plani ismini tutar.
- **incomingContext:** PBX'in gelen çağrılar için tanımlanmış çağrı plani ismini tutar.
- **internalContext:** PBX'in dahili çağrılar için tanımlanmış çağrı plani ismini tutar.



Şekil 6.3 AsteriskPBXParameters sınıfı

6.2.3 DBParameters Sınıfı

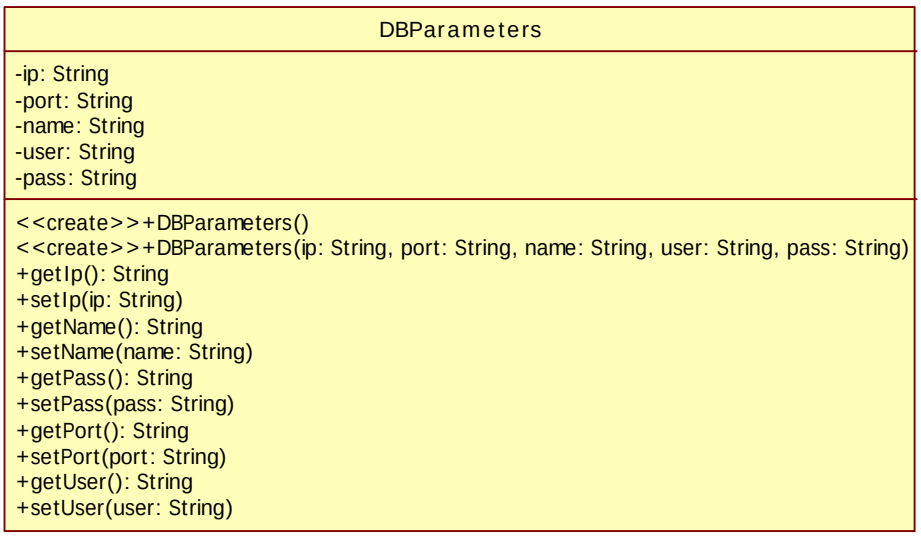
DBParameters, veritabanı ile kurulacak bağlantı için gerekli tanımları tutan sınıftır.

DBParameters sınıfı 12 metot içermektedir (Şekil 6.4). Bu metodlar şu şekilde sıralanmaktadır:

- **DBParameters:** Sınıfın construtor metodudur. Sınıfın null değerlere sahip özelliklerden oluşan boş bir kopyasını oluşturur.
- **DBParameters:** Sınıfın ikinci constructor metodudur. Sınıfın parametre olarak verilen değerlere sahip özelliklerinden oluşan bir kopyasını oluşturur.
- **getIP:** Veritabanı bağlantısı için kullanılan IP adresini verir.
- **setIP:** Veritabanı bağlantısı için kullanılan IP adresini kaydeder.
- **getName:** Sorgularda kullanılan veritabanı adını verir.
- **setName:** Sorgularda kullanılan veritabanı adını kaydeder.
- **getPass:** Veritabanı bağlantısı için kullanılan şifreyi verir.
- **setPass:** Veritabanı bağlantısı için kullanılan şifreyi kaydeder.
- **getPort:** Veritabanı bağlantısı için kullanılan port numarasını verir.
- **setPort:** Veritabanı bağlantısı için kuıllanılan port numarasını kaydeder.
- **getUser:** Veritabanı bağlantısı için kullanılan kullanıcı adını verir.
- **setUser:** Veritabanı bağlantısı için kullanılan kullanıcı adını kaydeder.

DBParameters sınıfı 5 adet özellik içermektedir (Şekil 6.4). Bu özellikler şunlardır:

- **ip:** Bağlanılan veritabanının IP adresini tutar.
- **port:** Bağlanılan veritabanı sunucusunun bağlantı portunu tutar.
- **name:** Sunucu üzerinde kullanılan veritabanının adını tutar.
- **user:** Veritabanı bağlantısında kullanılan kullanıcı adını tutar.
- **pass:** Veritabanı bağlantısında kullanılan şifreyi tutar.



Şekil 6.4 DBParameters sınıfı

Farklı veritabanlarına bağlantı için oluşturulacak ve ek özellikler içerecek sınıflar DBParameters sınıfı genişletir.

6.2.4 PBXCall Sınıfı

PBXCall, PBX üzerinde gerçekleşen bir çağrının bilgilerini tutan sınıftır.

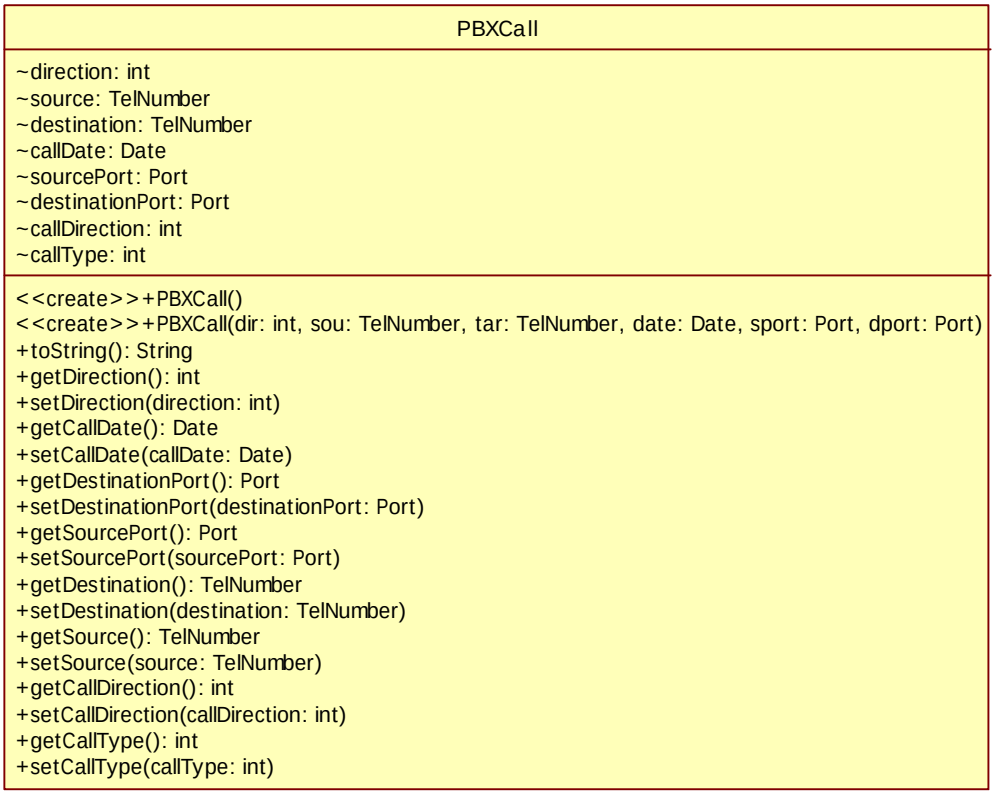
PBXCall sınıfı 19 metot içermektedir (Şekil 6.5). Bu metodlar şu şekilde sıralanmaktadır:

- **PBXCall:** Sınıfın constructor metodudur. Sınıfın null değerlere sahip özelliklerden oluşan boş bir kopyasını oluşturur.
- **PBXCall:** Sınıfın ikinci constructor metodudur. Sınıfın verilen değerlere sahip özelliklerden oluşan bir kopyasını oluşturur.
- **toString:** Çağrının özelliklerinden oluşan özet bir bilgi metni oluşturur.
- **getDirection:** Çağrının giden, gelen veya dahili tipteki yönünü verir.
- **setDirection:** Çağrının giden, gelen veya dahili tipteki yönünü kaydeder
- **getCallDate:** Çağrının başlangıç tarih ve saatini verir.
- **setCallDate:** Çağrının başlangıç tarih ve saatini kaydeder.
- **getDestinationPort:** Çağrıda aranan portu verir.
- **setDestinationPort:** Çağrıda aranan portu tanımlar.

- **getSourcePort:** Çağrıda arayan portu verir.
- **setSourcePort:** Çağrıda arayan portu verir.
- **getDestination:** Aranan numarayı verir.
- **setDestination:** Aranan numarayı tanımlar.
- **getSource:** Arayan numarayı verir.
- **setSource:** Arayan numarayı tanımlar.
- **getCallDirection:** Çağrının şehiriçi, şehirlarası, GSM türünden arama yönünü verir.
- **setCallDirection:** Çağrının şehiriçi, şehirlerarası, GSM türünden arama yönünü tanımlar.
- **getCallType:** Çağrının, sistem rehberine göre belirlenen arama tipini verir.
- **setCallType:** Çağrının arama tipini tanımlar.

PBXCall sınıfı 8 adet özellik içermektedir. Bunlar:

- **direction:** çağrının giden, gelen veya dahili olmak üzere yönünü tutar.
- **source:** çağrıyı başlatan numarayı tutar.
- **destination:** çağrının hedef numarasını tutar.
- **callDate:** çağrının başlatıldığı tarih ve saati tutar.
- **sourcePort:** çağrıyı başlatan kaynak portu tutar.
- **destinationPort:** çağrının hedef portunu tutar.
- **callDirection:** çağrının şehiriçi, şehirlerarası ve GSM olmak üzere arama yönünü tutar.
- **callType:** çağrının sistem rehberine göre belirlenmiş arama tipini tutar.



Şekil 6.5 PBXCall sınıfı

6.2.5 Port Sınıfı

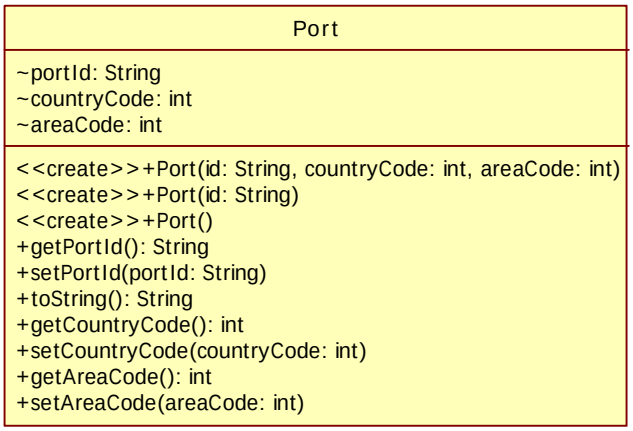
Port, PBX üzerindeki hat bilgisinin tutulduğu sınıftır.

Port sınıfı 10 metot içermektedir (Şekil 6.6). Bu metodlar şunlardır:

- **port:** Sınıfın constructor metodudur. Sınıfın null değerlere sahip özelliklerden oluşan boş bir kopyasını oluşturur.
- **port:** Sınıfın ikinci constructor metodudur. Sınıfın parametre olarak verilen port ismi ve null değere sahip ülke ve alan kodu özelliklerinden oluşan bir kopyasını oluşturur.
- **Port:** Sınıfın üçüncü constructor metodudur. Sınıfın parametre olarak verilen değerlere sahip özelliklerden oluşan bir kopyasını oluşturur.
- **getPortId:** portun özel ismini verir.
- **setPortId:** portun özel ismini kaydeder.
- **toString:** portun özellik ve değerlerini anlamlı ve özet şekilde verir.
- **getCountryCode:** portun bağlı olduğu hattın ülke kodunu verir.
- **setCountryCode:** portun bağlı olduğu hattın ülke kodunu kaydeder.
- **getAreaCode:** portun bağlı olduğu hattın alan kodunu verir.
- **setAreaCode:** portun bağlı olduğu hattın alan kodunu kaydeder.

Port sınıfı 3 adet özellik içermektedir (Şekil 6.6). Bu özellikler şu şekilde sıralanmaktadır:

- **portId:** portun özel ismini tutar.
- **countryCode:** portun bağlı bulunduğu hattın ülke kodunu tutar.
- **areaCode:** portun bağlı bulunduğu hattın alan kodunu tutar.



Şekil 6.6 Port sınıfı

6.2.6 Rule Sınıfı

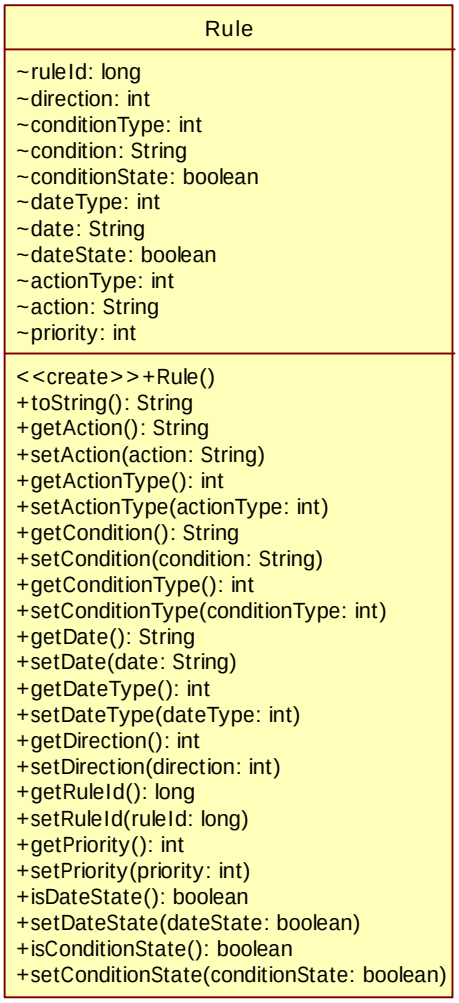
Rule, sistemin karar mekanzimasının gerçekleşen olayları değerlendirirken kullandığı kural bilgisinin tutulduğu sınıftır.

Rule sınıfı 24 metot içermektedir (Şekil 6.7). Bu metodlar şu şekilde sıralanmaktadır:

- **rule:** sınıfın constructor metodudur. Sınıfın null değerlere sahip özelliklerden oluşan boş bir kopyasını oluşturur.
- **toString:** sınıfın özellik değerlerinden oluşan özet bir metni verir.
- **getAction:** kuralın geçerli olması durumunda yerine getirilecek işlem parametrelerini verir.
- **setAction:** kuralın geçerli olması durumunda yerine getirilecek işlem parametrelerini kaydeder.
- **getActionType:** kuralın geçerli olması durumunda yerine getirilecek işlemin tipini verir.
- **setActionType:** kuralın geçerli olması durumunda yerine getirilecek işlemin tipini kaydeder.
- **getCondition:** kuralın geçerli olması durumunu tanımlayan parametreleri verir.
- **setConditionType:** kuralın geçerli olması durumunu tanımlayan parametreleri kaydeder.
- **getConditionType:** kuralın geçerli olma durum tipini verir.
- **setConditionType:** kuralın geçerli olma durum tipini kaydeder.
- **getDate:** kuralın geçerli olacağı zaman parametresini verir.
- **setDate:** kuralın geçerli olacağı zaman parametresini kaydeder.
- **getDateType:** kuralın geçerli olacağı zaman tipini verir.
- **setDateType:** kuralın geçerli olacağı zaman tipini kaydeder.
- **getDirection:** kuralın geçerli olacağı arama yönünü verir.
- **setDirection:** kuralın geçerli olacağı arama yönünü kaydeder.
- **getRuleId:** kuralın özel ismini verir.
- **setRuleId:** kuralın özel ismini kaydeder.
- **getPriority:** kuralın öncelik değerini verir.
- **setPriority:** kuralın öncelik değerini kaydeder.
- **isDateState:** kuralın geçerli olacağı tarihin tersine çevrilme durumunu verir.
- **setDateState:** kuralın geçerli olacağı tarihin tersine çevrilmesini belirler.
- **isConditionState:** kuralın geçerli olma durumunu tersine çevrilme halini verir.
- **setConditionState:** kuralın geçerli olma durumunu tersine çevrilme halini belirler.

Rule sınıfı 11 adet özellik içermektedir (Şekil 6.7). Bu özellikler şunlardır:

- **ruleId:** kuralın özel ismini tutar.
- **direction:** kuralın uygulanacağı arama yönünü tutar.
- **conditionType:** kuralın geçerli olma durum tipini tutar.
- **condition:** kuralın geçerli olma durumunun parametrelerini tutar.
- **conditionState:** kuralın geçerli olm durumunun tersine çevrilme şartını tutar.
- **dateType:** kuralın geçerli olduğu zaman birimini tutar.
- **date:** kuralın geçerli olduğu zaman parametrelerini tutar.
- **dateState:** kuralın geçerli olduğu zamanı tersine çevirme durumunu tutar.
- **actionType:** kuralın geçerli olması durumunda gerçekleştirilecek işlemin tipini tutar.
- **action:** kuralın geçerli olması durumunda gerçekleştirilecek işlemin parametrelerin tutar.
- **priority:** kuralın öncelik değerini tutar.



Şekil 6.7 Rule sınıfı

6.2.7 TelNumber Sınıfı

TelNumber, sistemde kayıtlı telefon numaralarının saklandığı ve taşındığı sınıftır.

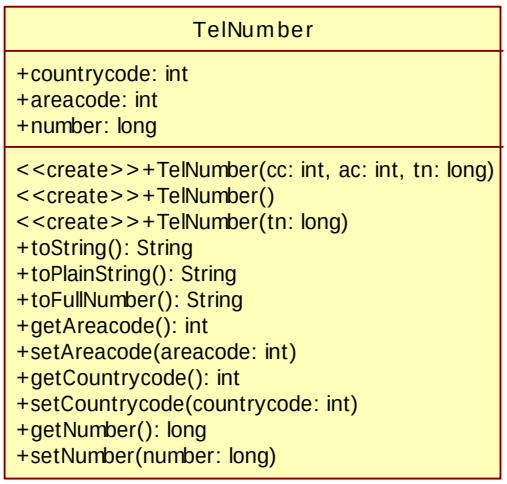
TelNumer sınıfı 12 metot içermektedir (Şekil 6.8). Bu metodlar şu şekilde sıralanmaktadır:

- **TelNumber:** Sınıfın constructor metodudur. Sınıfın parametre olarak verilen ülke kodu, alan kodu ve telefon numarası değerlerine sahip özelliklerden oluşan bir kopyasını oluşturur.
- **TelNumber:** Sınıfın ikinci constructor metodudur. Sınıfın ön tanımlı değerlere sahip özelliklerden oluşan boş bir kopyasını oluşturur.
- **TelNumber:** Sınıfın üçüncü constructor metodudur. Sınıfın ön tanımlı ülke ve alan koduna ve parametre olarak verilen telefon numarası değerlerine sahip özelliklerden oluşan bir kopyasını oluşturur.
- **toString:** Sınıfın tanımlı özelliklerinin değerlerini formatlı bir metin halinde verir.
- **toPlainString:** sadece tanımlı özelliklerden oluşan nümerik yapıda bir telefon numarası metni verir.

- **toFullNumber:** Sınıfın tüm özelliklerinin değerlerini formatlı bir metin halinde verir.
- **getAreacode:** Numaranın alan kodunu verir.
- **setAreacode:** Numaranın alan kodunu kaydeder.
- **getCountrycode:** Numaranın ülke kodunu verir.
- **setCountrycode:** Numaranın ülke kodunu kaydeder.
- **getNumber:** Telefon abone numarasını verir.
- **setNumber:** Telefon abone numarasını kaydeder.

TelNumber sınıfında 3 adet özellik bulunmaktadır (Şekil 6.8). Bu özellikler şunlardır:

- **countrycode:** numaranın ülke kodunu tutar.
- **areacode:** numaranın alan kodunu tutar.
- **number:** numaranın abone numarasını tutar.



Şekil 6.8 TelNumber sınıfı

6.3 SQL Paketi

SQL paketi, SQL tabanlı veritabanlarına bağlantı kurmak ve bu veritabanlarıyla yapılacak işlemleri gerçekleştirecek sınıf ve metodların yer aldığı pakettir.

SQL paketindeki veri kaynağı sınıfları, SQLDataSource arayüzünü gerçeklemek zorundadır. Bu sayede diğer sınıfların SQLDataSource arayüzü üzerinden farklı veritabanlarına bağlanabilmesi sağlanır.

Gerçeklenen uygulamada kullanılan Microsoft SQL Server ile çalışabilmek üzere MSSQLServerDBSource sınıfı SQL paketi içinde gerçeklenmiştir.

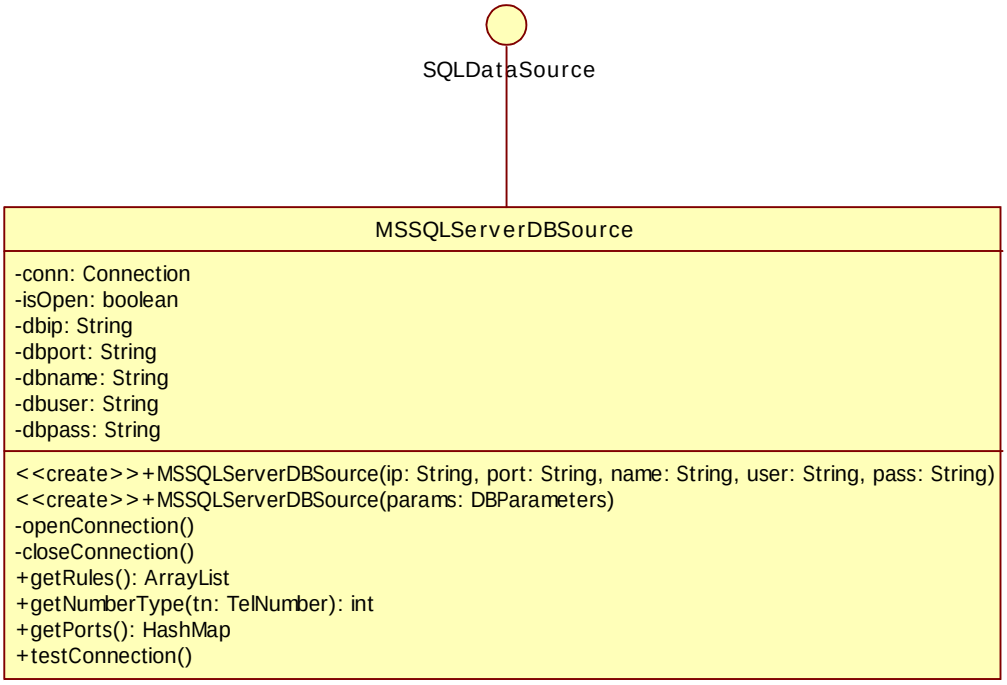
6.3.1 MSSQLServerDBSource Sınıfı

MSSQLServerDBSource sınıfı, sistemin Microsoft SQL Server ile bağlantı kurmasını ve bu bağlantı üzerinden komutlar çalıştırılmasına imkan sağlayan metot ve özellikleri içermektedir.

MSSQLServerDBSource, SQLDataSource arayüzünü implemente etmiştir. Böylece bu sınıfa da diğer SQL kaynaklarıyla aynı arayüzden erişim imkanı tanınmıştır.

MSSQLServerDBSource sınıfında 8 method tanımlanmıştır (Şekil 6.9). Bu methodlar:

- **MSSQLServerDBSource:** Sınıfın constructor metodudur. Sınıfın parametre olarak verilen değerlere sahip özelliklerle bir kopyasını oluşturur.
- **MSSQLServerDBSource:** Sınıfın ikinci constructor metodudur. Parametre olarak DBParameters tipinde bir obje alır ve bu objede tanımlanan değerlere sahip özelliklerle bir kopya oluştuur.
- **openConnection:** Veritabanıyla bağlantıyı kurar.
- **closeConnection:** Veritabanı bağlantısını sonlandırır.
- **getRules:** Veritabanından kural listesini alır.
- **getNumberType:** Veritabanından bir numaranın tipini alır.
- **getPorts:** Veritabanından sistem portlarının listesini alır.
- **testConnection:** Veritabanı bağlantısının durumunu kontrol eder.



Şekil 6.9 MSSQLServerDBSoure sınıfı

MSSQLServerDBSource sınıfında 7 özellik bulunmaktadır. Bu özellikler şu şekilde sıralanmaktadır:

- **conn:** Veritabanıyla kurulmuş bağlantı objesini tutar. Veritabanı komutları bu obje üzerinden işletilir.
- **isOpen:** Veritabanı bağlantısının durumunu tutar.
- **dbip:** Veritabanı bağlantısında kullanılan IP adresini tutar.
- **dbport:** Veritabanı bağlantısında kullanılan port adresini tutar.
- **dbname:** Veritabanı sunucusunda kullanılan veritabanının ismini tutar.
- **dbuser:** Veritabanı bağlantısında kullanılan kullanıcı ismini tutar.
- **dbpass:** Veritabanı bağlantısında kullanılan şifreyi tutar.

6.4 Judgement Paketi

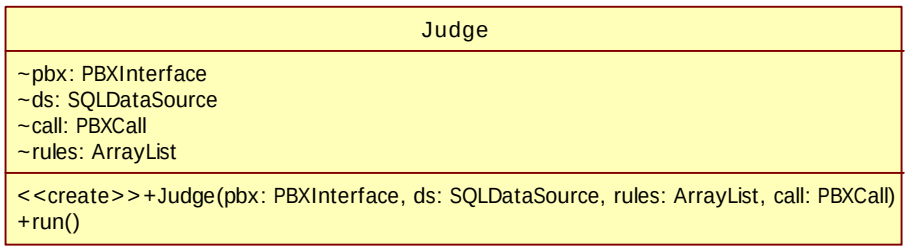
Judgement paketi, uygulamanın karar mekanizmasıyla ilgili sınıflarını içerir. Farklı tipteki PBX olaylarının veya sistem durumu değişikliklerinin değerlendirilmesi için gerçeklenmiş metodlar Judgement paketi içinde yer almaktadır.

Gerçeklenen uygulamada Judgement paketi içinde Judge sınıfı yer almaktadır. Threaded yapıda gerçeklenmiş bu sınıf, ana uygulama tarafından farklı PBX olaylarının değerlendirilmesi için birden çok sayıda oluşturulabilmektedir.

6.4.1 Judge Sınıfı

Judge sınıfı, sistemde başlatılan bir çağrıyı aktarılan kural listesine göre değerlendirir ve yerine getirilmesi gereken işlemi PBX arayüzüne iletir. Kendine ait bir thread içinde çalışan Judge objesi, görevini bitirdikten sonra sonlanır.

Judge sınıfı bir constructor ve run metodundan oluşmaktadır (Şekil 6.10). Sınıf oluşturulduktan sonra run metodundaki işlemler yerine getirilir ve akış sonucu objenin varlığı sonlandırılır.



Şekil 6.10 Judge sınıfı

Judge sınıfı 4 adet özellik içermektedir. Bunlar şu şekilde sıralanmaktadır:

- **pbx:** işlemin gönderileceği PBX arayüzünü tutar.
- **ds:** değerlendirme işlemi sırasında kullanılabilecek veri kaynağı arayüzünü tutar.
- **call:** değerlendirilecek çağrının bilgisini tutar.
- **rules:** değerlendirmede kullanılan kural listesini tutar.

6.5 PBX Paketi

PBX paketi, uygulamanın farklı PBX'ler ile çalışabilmesini sağlamak üzere gerçeklenmiş sınıfları ve sistemin bu sınıflarla iletişim kurmasını sağlayan metodları içermektedir.

PBX paketinde yer alan PBX yöneticisi sınıfları, PBXManager sınıfını genişletmektedir. Bu sınıf, tüm yönetici sınıflarda bulunması gereken temel metot ve özellikleri kapsar. PBXManager sınıfı PBXInterface arayüzünü implemente eder. Böylece diğer sınıfların PBX yöneticilerine tek bir arayüzden erişimi sağlanmıştır.

PBX paketindeki sınıflar şu şekilde sıralanmaktadır:

- PBXManager
- AsteriskPBXManager
- PBXListener

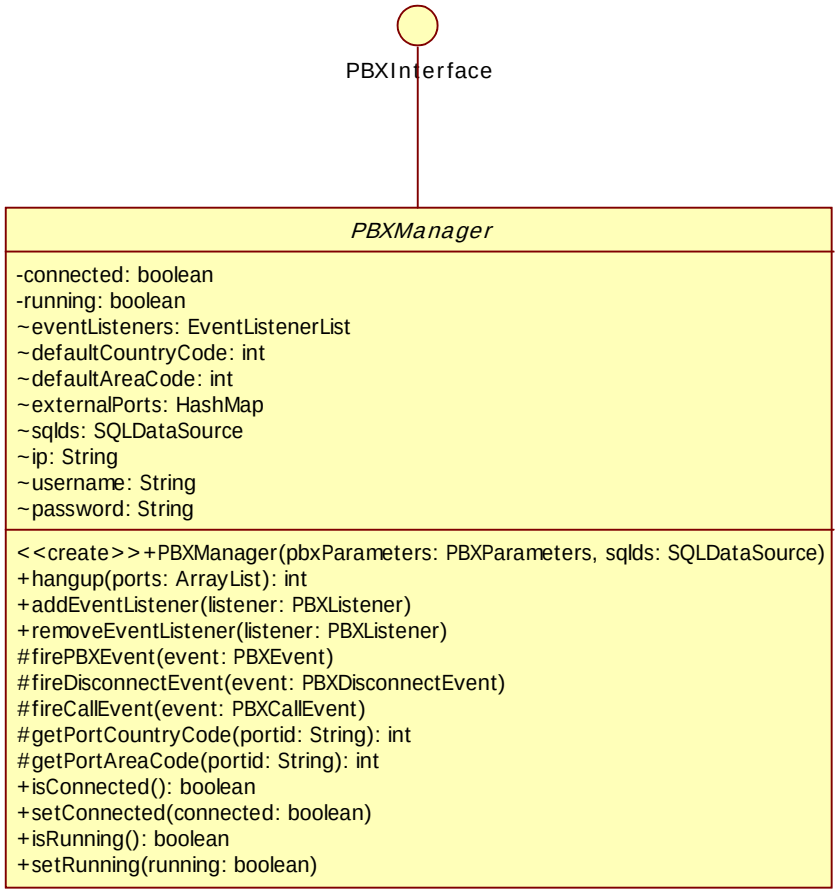
6.5.1 PBXManager Sınıfı

PBXManager sınıfı, farklı tipteki PBX'lerle çalışmak üzere gerçeklenen PBX yöneticisi sınıflarının genişlettiği, tüm PBX yöneticisi sınıflarında temel olarak olması gereken metot ve özelliklerin tanımlandığı sınıftır. PBXManager sınıfı PBXInterface arayüzünü implemente ederek PBX yöneticisi sınıflarını kullanacak diğer modüller için tek ve temel bir yapı ortaya koymaktadır.

PBXManager sınıfında 13 metot tanımlanmıştır (Şekil 6.11). Bu metodlar şunlardır:

- **PBXManager:** Sınıfın constructor metodudur. Parametre olarak verilen değerlere sahip özelliklerle sınıfın bir kopyasını oluşturur.
- **hangup:** Birden fazla sayıda port için arayüz üzerinde tanımlanmış hangup komutunu çalıştırır.
- **addEventListener:** PBX olaylarının iletileceği yeni bir dinleyici tanımlar.
- **removeEventListener:** PBX olaylarının iletildiği bir dinleyiciyi devre dışı bırakır.
- **firePBXEvent:** PBX üzerinde gerçekleşen bir olayı dinleyicilere gönderir.
- **fireDisconnecEvent:** PBX üzerindeki bir çağrı sonlandırılma olayını dinleyicilere gönderir.
- **fireCallEvent:** PBX üzerindeki bir çağrı başlatma olayını dinleyicilere gönderir.

- **getPortCountryCode:** Veri kaynağından portun bağlı olduğu hattın ülke kodunu alır.
- **getPortAreaCode:** Veri kaynağından portun bağlı olduğu hattın alan kodunu alır.
- **isConnected:** PBX bağlantısının durumunu verir.
- **setConnected:** PBX bağlantısının durumunu değiştirir.
- **isRunning:** Olay iletiminin başlatılma durumunu verir.
- **setRunning:** Olay iletiminin başlatılma durumunu değiştirir.



Şekil 6.11 PBXManager sınıfı

PBXManager sınıfında 10 özellik tanımlanmıştır (Şekil 6.11). Bu özellikler şunlardır:

- **connected:** PBX'in bağlantı durumunu tutar.
- **running:** Olay iletiminin başlatılma durumunu tutar.
- **eventListeners:** Olay iletiminin yapılacağı dinleyicilerin listesini tutar.
- **defaultCountryCode:** Tanımsız portlar için kullanılacak ön tanımlı ülke kodunu tutar.
- **defaultAreaCode:** Tanımsız portlar için kullanılacak ön tanımlı alan kodunu tutar.
- **externalPorts:** Sistemdeki dış portların listesini tutar.
- **sqlds:** Veri kaynağı objesini tutar.
- **ip:** PBX'e bağlantı için kullanılacak IP adresini tutar.
- **username:** PBX'e bağlantı için kullanılacak kullanıcı adını tutar.
- **password:** PBX'e bağlantı için kullanılacak şifreyi tutar.

6.5.2 AsteriskPBXManager Sınıfı

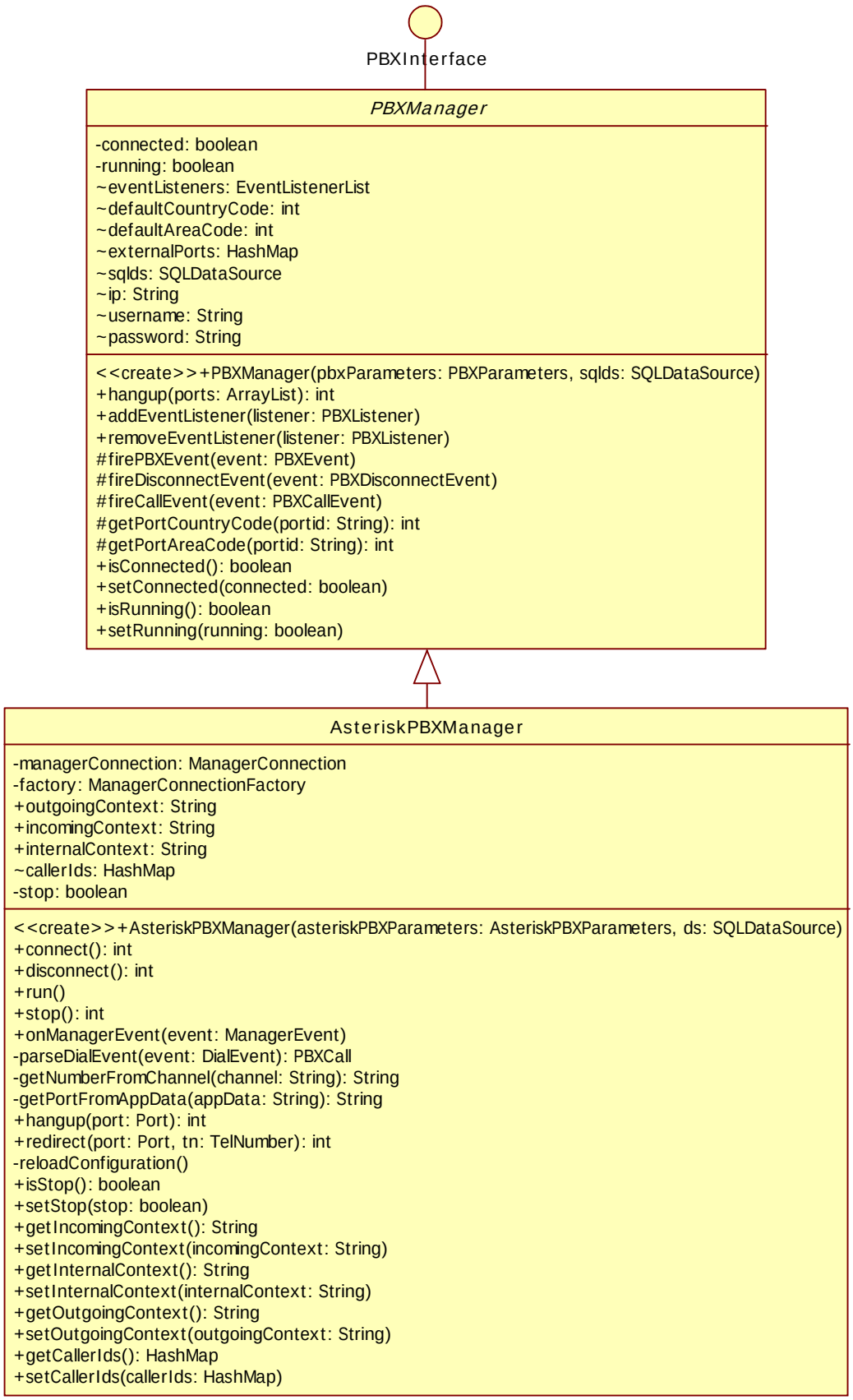
AsteriskPBXManager, Asterisk PBX'e bağlantı kurmak ve bu PBX üzerinde çağrı başlatma, sonlandırma ve yönlendirme gibi işlemler gerçekleştirmek üzere gerçeklenmiş bir PBX yöneticisi sınıfıdır.

AsteriskPBXManager, PBXManager sınıfını genişletmekte ve bu sınıfın tüm özelliklerini kapsamaktadır. PBXManager, PBXInterface arayüzünü implemente ettiğinden AsteriskPBXManager da bu arayüzü implemente etmektedir. Bu sayede diğer sınıflar AsteriskPBXManager'a da PBXInterface arayüzünden erişebilmekte ve standart PBX işlemlerini Asterisk üzerinde gerçekleştirebilmektedir.

AsteriskPBXManager threaded yapıda gerçeklenmiştir. Kendisini oluşturan sınıftan bağımsız olarak çalışmasını sürdürmektedir. Diğer sınıflarla iletişim dinleyiciler üzerinden gerçekleştirilir.

AsteriskPBXManager dahilinde, genişletilen sınıf metodları dışında 13 metot bulunmaktadır (Şekil 6.12). Bu metodlar şu şekilde sıralanmaktadır:

- **AsteriskPBXManager:** Sınıfın constructor metodudur. Parametre olarak verilen değerlere sahip özelliklerden oluşan bir sınıf oluşturur.
- **connect:** Asterisk'e olan bağlantıyı kurar.
- **disconnect:** Asterisk bağlantısını sonlandırır.
- **run:** Olay iletimini başlatır.
- **stop:** Olay iletimini sonlandırır.
- **onManagerEvent:** PBX üzerinden gelen olayları sistemdeki tanımlı yapılara dönüştürür.
- **parseDialEvent:** PBX üzerinde gerçekleşen bir çağrı olayını işler ve tanımlamaları yapar.
- **getNumberFromChannel:** Çağrı kanalı üzerindeki numara bilgisini alır.
- **getPortFromAppData:** Çağrı bilgisi üzerindeki port ismini alır.
- **hangup:** Çağrıyı sonlandırır.
- **redirect:** Çağrıyı başka bir porta yönlendirir.
- **reloadConfiguration:** Asterisk tanımlamalarını tekrar yükler.
- **isStop:** Olay iletimi durumunu verir.
- **setStop:** Olay iletimi durumunu değiştirir.
- **getIncomingContext:** Asterisk üzerindeki gelen çağrılar arama planı ismini alır.
- **setIncomingContext:** Asterisk üzerindeki gelen çağrılar arama planı ismini tanımlar.
- **getInternalContext:** Asterisk üzerindeki dahili çağrılar arama planı ismini alır.
- **setInternalContext:** Asterisk üzerindeki dahili çağrılar arama planı ismini tanımlar.
- **getOutgoingContext:** Asterisk üzerindeki giden çağrılar arama planı ismini alır.
- **setOutgoingContext:** Asterisk üzerindeki giden çağrılar arama planı ismini tanımlar.
- **getCallerIds:** Kanal-Numara-Açıklama eşleşmelerini alır.
- **setCallerIds:** Kanal-Numara-Açıklama eşleşmelerini tanımlar.



Şekil 6.12 AsteriskPBXManager sınıfı

AsteriskPBXManager sınıfı dahilinde tanımlanmış 7 özellik bulunmaktadır (Şekil 6.12). Bu özellikler şunlardır:

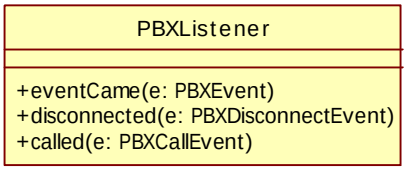
- **managerConnection:** Asterisk PBX ile olan bağlantı objesini tutar. PBX üzerindeki işlemler bu obje üzerinden gerçekleştirilir.
- **factory:** Asterisk PBX ile olan bağlantının kurulması için kullanılan objeyı tutar.
- **outgoingContext:** Asterisk PBX üzerindeki giden çağrı arama planı ismini tutar.
- **incomingContext:** Asterisk PBX üzerinde gelen çağrı arama planı ismini tutar.
- **internalContext:** Asterisk PBX üzerinde iç çağrı arama planı ismini tutar
- **callerIds:** Kanal-numara-açıklama eşleşme listesini tutar.
- **stop:** Olayların iletilme durumunu tutar.

6.5.3 PBXListener Sınıfı

PBXListener, PBX olaylarının PBX yöneticisi sınıfından diğer sınıflara iletilmesi ve tanımlanan işlemlerin yapılması için kullanılan sınıftır. PBX olayları sonucu işlem gerçekleştirmek isteyen modüller, bu sınıfın bir kopyasını override eder ve ilgili PBX yöneticisine kaydettirir. Bu aşamada PBX yöneticisi bir PBX olayı gerçekleştiğinde dinleyici tarafından tanımlanmış metodları tetikler.

PBXListener sınıfı üç metot içermektedir. Bu metodlar şu şekilde sıralanmaktadır:

- **eventCame:** Herhangi bir PBX olayı geldiğinde gerçekleştirilecek işlemleri yerine getirir.
- **disconnected:** Çağrı sonlandırılma olayı sonucu gerçekleştirilecek işlemleri yerine getirir.
- **called:** Çağrı başlatılması sonucu gerçekleştirilecek işlemleri yerine getirir.



Şekil 6.13 PBXListener sınıfı

6.6 Events Paketi

Events paketi, farklı tipteki PBX olay bilgilerinin tutulduğu sınıfları içermektedir. Events paketinde 3 sınıf bulunmaktadır. Bu sınıflar:

- PBXEvent
- PBXCallEvent

- PBXDisconnectEvent

6.6.1 PBXEvent Sınıfı

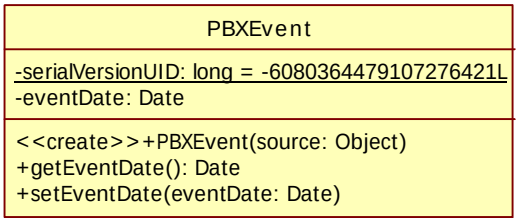
PBXEvent, PBX'ten herhangi bir olay gerçekleştiğinde oluşturulan ve iletilen sınıftır. Olayın gerçekleştiği kaynak ve olayla ilgili genel bilgiyi içerir. Tüm olay sınıfları bu sınıfı genişletmektedir.

PBXEvent sınıfında 3 metot tanımlanmıştır (Şekil 6.14). Bu metodlar şunlardır:

- **PBXEvent:** Sınıfın constructor metodudur. Oluşturulma zamanı ve kaynak değerlerinden oluşan bir sınıf kopyası ortaya koyar.
- **getEventDate:** Olayın gerçekleşme zamanını verir.
- **setEventDate:** Olayın gerçekleşme zamanını kaydeder.

PBXEvent sınıfında 2 özellik tanımlanmıştır (Şekil 6.14). Bu özellikler şunlardır:

- **serialVersionUID:** Sınıfın serileştirme işlemine girmesi durumunda kullanılan numaradır.
- **eventDate:** Olayın gerçekleşme zamanını tutar.



Şekil 6.14 PBXEvent sınıfı

6.6.2 PBXCallEvent Sınıfı

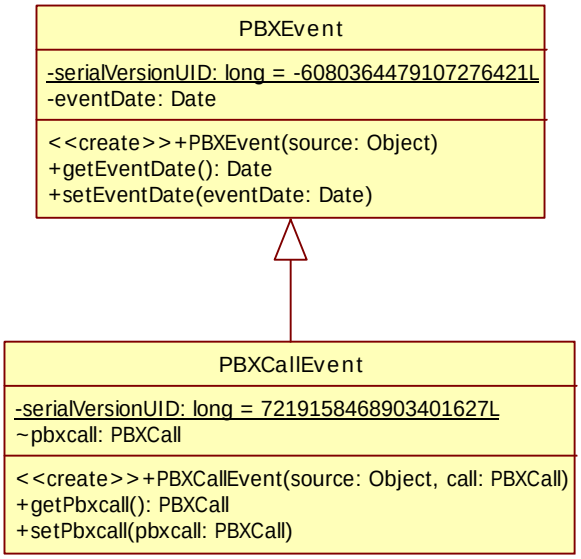
PBXCallEvent, PBX üzerinde başlatılan bir çağrı olduğunda oluşturulan olayın bilgisini tutan sınıftır. PBXCallEvent sınıfı, PBXEvent sınıfının genişletilmesiyle oluşturulmuştur ve bu sınıfın metot ve özelliklerini de kapsar.

PBXCallEvent dahilinde 3 metot tanımlanmıştır (Şekil 6.15). Bu metodlar şunlardır:

- **PBXCallEvent:** Sınıfın constructor metodudur. Kaynak obje ve başlatılan çağrı bilgisi parametrelerinden bu değerlere sahip özelliklerle sınıfın bir kopyasını oluşturur.
- **getPbxcall:** Olayı tetikleyen çağrı bilgisini verir.
- **setPbxcall:** Olayı tetikleyen çağrı bilgisini kaydeder.

PBXCallEvent sınıfında 2 özellik tanımlanmıştır (Şekil 6.15). Bu özellikler şunlardır:

- **serialVersionUID:** Sınıfın serileştirme işlemine girmesi sırasında kullanılan numaradır.
- **pbxcall:** Olayı tetikleyen çağrının bilgilerini tutar.



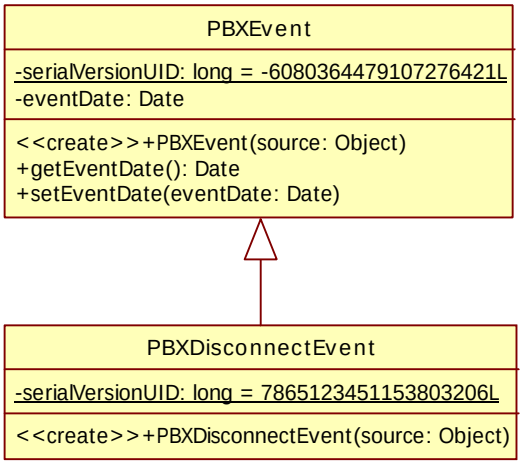
Şekil 6.15 PBXCallEvent sınıfı

6.6.3 PBXDisconnectEvent Sınıfı

PBXDisconnectEvent, PBX üzerinde sonlandırılan bir çağrı olduğunda oluşturulan olayın bilgisini tutan sınıftır. PBXDisconnectEvent sınıfı, PBXEvent sınıfının genişletilmesiyle oluşturulmuştur ve bu sınıfın metot ve özelliklerini de kapsar.

PBXCallEvent dahilinde tek metot tanımlanmıştır (Şekil 6.16). Sınıfın constructoru olan bu metot parametre olarak verilen kaynak objeden bir sınıf kopyası oluşturur.

PBXCallEvent sınıfında tek özellik tanımlanmıştır (Şekil 6.15). Bu özellik sınıfın serileştirme işlemine girmesi sırasında kullanılır.



Şekil 6.16 PBXDisconnectEvent sınıfı

6.7 Util Paketi

Util paketi, sistem çapında ihtiyaç duyulan genel ve özel amaçlı fonksiyonları amaca yönelik olmak üzere belirli gruplar halinde toplayan sınıfları içermektedir. Bu paket dahilindeki sınıflar birden çok modül tarafından ihtiyaç duyulabilecek fonksiyonları gerçekleyerek uygulamadaki kod tekrarını düşürmektedir.

Gerçekleştirilen uygulama dahilinde Util paketinde 2 sınıf bulunmaktadır. Bu sınıflar şunlardır:

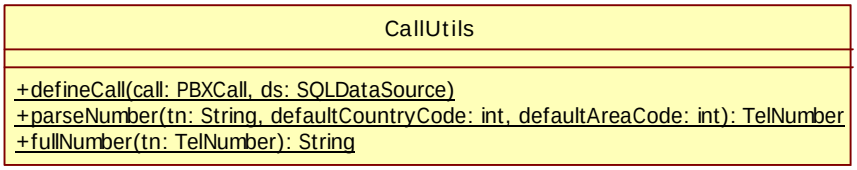
- CallUtils
- RuleUtils

6.7.1 CallUtils Sınıfı

CallUtils, çağrı bilgisini taşıyan objeler veya çağrı özellikleri üzerinde gerçekleştirilebilecek metodları gerçekleyen sınıftır.

CallUtils sınıfı 3 metot gerçeklemektedir (Şekil 6.17). Bu metodlar şunlardır:

- **defineCall:** Bir çağrının veri kaynağındaki tanımlar çerçevesinde yön ve tip bilgisinin tanımlanmasını sağlar.
- **parseNumber:** Bir metin parçasındaki numaranın sistem telefon numarası objesine çevrimini sağlar.
- **fullNumber:** Bir telefon numarası objesinden formatlı telefon numarası oluşturur.

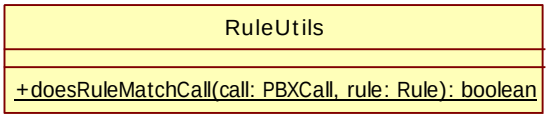


Şekil 6.17 CallUtils sınıfı

6.7.2 RuleUtils Sınıfı

RuleUtils, kural bilgisini taşıyan objeler veya kural özellikleri üzerinde gerçekleştirilebilecek metodları gerçekleyen sınıftır.

Gerçeklenen uygulamada RuleUtils sınıfı tek bir metot içermektedir (Şekil 6.18). Bu metod, bir çağrıyla bir kural nesnesini karşılaştırır ve eşleşme olup olmadığını değerlendirir.



Şekil 6.18 RuleUtils sınıfı

7. UYGULAMA SENARYOLARI

Tez çalışmasının bu bölümünde giriş bölümünde tanımlanan, özgün değer ve yaygın etki bölümünde detaylandırılan ihtiyaç ve amaçlar çerçevesinde tasarlanan ve gerçeklenen ses haberleşme sistemleri erişim kontrolü sisteminin kullanım alanları ve elde edilecek sonuçlar değerlendirilmektedir.

Günümüz ses haberleşme sistemleri ve tasarlanan sistemin yapısı ile yetenekleri ele alındığında, başlıca uygulama alanlarını beş grupta toplamak mümkündür. Bunlar;

- Kurumlarda görüşme disiplini arttırma uygulamaları
- Ön ödemeli servisler ile katma değer uygulamaları
- Numara taşınabilirliği uygulamalarında tasarruf sağlanması
- Çağrı yönlendirme temelli verimlilik uygulamaları
- Spam çağrı engellemeye dayalı verimlilik uygulamaları

olarak sıralanmaktadır.

7.1 Görüşme Disiplini Uygulamaları

Giderlerin aşırı yükselmesini önlemek ve zamanı verimli kullanabilmek amacıyla iş için gerekli olmayan telefon görüşmelerinin kontrol altına alınmasına görüşme disiplini adı verilir.

Telefon görüşmelerinin hiç kontrol edilmediği kurumlarda çalışanlar, istediği zaman istediği telefon görüşmesini yapabilmekte, bu kurumlardaki yöneticiler serbestliğin motivasyonu ve iş performansını arttırdığını düşünmektedir. Fakat, telefon görülmelerinin hiç kontrol edilmiyor olması da önemli sakıncalara neden olabilmektedir. Yapılan araştırmalar, belirli bir görüşme disiplini politikası geliştirmemiş şirketlerde, iş dışı amaçlarla yapılan görüşmelerin maliyetinin toplam telefon maliyetinin ortalama %17'sini oluşturduğunu ve bu oranın %40'a kadar yükselebildiğini göstermiştir [25]. İş dışı görüşmelerin getirdiği maliyet genel iletişim maliyeti içinde önemli bir oran oluşturmadığı durumlarda dahi kurum içi iş disipini açısından sakıncalar ortaya çıkmaktadır. Aşırı oranda iş dışı görüşmesi yapan bir çalışanın hiçbir sınırlama ve uyarıyla karşılaşmaması, bu konuda dikkatli davranan çalışanlar üzerinde olumsuz etki bırakmakta ve motivasyonunu düşürmektedir.

Belirli bir görüşme disiplini politikası sürdüren kurumlarda da benzer bir tablo ortaya çıkmaktadır. Bu şirketlerde yapılan bir araştırma, toplam telefon maliyetinin %8'inin iş dışı amaçlarla yapılan görüşmelerin oluşturduğunu göstermiştir [25]. Aynı araştırmaya göre şirkete yapılan aramalardaki özel amaçlı görüşmelerin oranı ise %30 olarak belirlenmiştir. Bu

oranlar, mevcut görüşme disiplini oluşturma uygulamalarının maliyetleri azaltmasına rağmen iş performansı üzerinde önemli bir katkı sağlamadığını göstermektedir. Aramaları sınırlandırılmış kullanıcılar, kendilerini arattırarak özel görüşmelerini gerçekleştirmeye devam ettikleri belirlenmiştir. Bu sonuçlar, sadece dış aramaların sınırlandırılması ile hedeflenen iş performansı değerlerine ulaşılamayacağını göstermektedir.

Görüşme disiplini konusunun önemi, çalışan sayısıyla geometrik orantılı olarak yükselmektedir. Yapılan araştırmalar 50 ve üzeri çalışanın bulunduğu şirketlerde görüşme disiplini konusunda politikalar üretildiğinde önemli faydalar elde edilebileceğini göstermektedir [26]. Özellikle organizasyon yapısının sürekli büyüme gösterdiği veya eleman sirkülasyonunun yüksek olduğu şirketlerde çalışmaya yeni başlayan kişilerin kurum kültürünü ve alışkanlıklarını benimseyene kadar gerçekleştirebilecekleri aşırı telefon kullanımımın engellenmesi ihtiyacı, bu şirketlerdeki sağlıklı görüşme disiplini politikalarının önemini arttırmaktadır.

Günümüzde, yukarıda değinilen sakıncaları önlemek için bazı standart uygulamalar oluşturulmuştur. Bu uygulamaların başında, çalışanların arama yetkilerini kısıtlamak yer almaktadır. Buna göre şirket içindeki çalışanların bir bölümünün veya tamamının milletlerarası, GSM ve şehirlerarası arama yapması engellenmekte, bu aramaların yapılmasının gerektiği durumlarda yetkili bir kişinin telefonu bağlaması sonucu görüşme gerçekleştirilebilmektedir. Başkası aracılığıyla yapılan görüşmelerin kişisel görüşmeleri azaltacağı düşüncesiyle yapılan bu uygulama çoğu zaman iş amaçlı görüşmelerin başlatılmasında da zorluk oluşturmakta ve telefonun sürekli aracılar vasıtasıyla yapılmasından dolayı zamanın verimli kullanılmasını engelleyerek iş verimine olumsuz etki yapmaktadır.

Tez çalışması ile gerçekleştirilen sistem, bu tarz uygulamalarda ortaya çıkabilecek olumsuz yönleri önlerken amaçlanan faydayı da arttırmayı sağlamaktadır. Sistem, bir kullanıcıya ait tüm dahili telefonların sınırlandırılması yerine yapılan görüşmeler üzerinden sınırlandırma imkanı vermektedir. Belirli bir yöndeki tüm görüşmeler yerine belirli numaralar belirli zaman dilimlerinde engellenmekte, bu tanımlama kullanıcı bazında yapılabilmektedir. Tanımlamalar hem dış hem iç aramalar yönünde yapılabildiğinden standart uygulamalarda yetersiz kalınan durumlar da karşılanmaktadır. Ortaya konulan bu hassas yetkilendirme, telefonların bağlanması için aracılar belirlenmesine gerek bırakmamakta, bu da telefon süreçlerinin verimini arttırmaktadır. Ek olarak, kullanıcılar belirlenen oranda özel görüşmelerini gerçekleştirebildiklerinden çalışan motivasyonunun düşmesi de önlenmektedir.

Görüşme disiplini sağlamak için uygulanan diğer bir yaygın yöntem ise belirli dönemler sonunda çalışanlara görüşmelerinin listelerinin verilerek özel görüşmelerini belirlemelerinin istenmesidir. Otokontrol sağlama amaçlı bu yöntem zamanla uygulamanın kanıksanması sonucu etkisini yitirmekte ve görüşmelerin üzerinden geçen zamanın etkisiyle doğru beliremenin kullanıcılar tarafından yapılması zorlaşmaktadır. Bu da uygulamanın amaçlanan sonucu vermemesine neden olmaktadır.

Tez çalışması sonucu ortaya konulan sistem, kişilerin özel numaralarını sürekli olarak saklamakta ve yapılan görüşmeler sırasında belirlenen kısıtlar dahilinde kullanmaktadır. Böylece kullanıcıların her fatura dönemi sonunda tekrar tekrar aynı nuımaraları belirlemeleri önlenmekte, uygulamanın rutinleşmesi ve hedeflenen etkiyi kaybetmesi engellenmektedir.

Görüşme disiplini sağlamak için gerçekleştirilen bir diğer uygulama da tüm görüşmelerin belirli bir süre sonunda otomatik olarak kesilmesidir. Bu uygulama, telefon giderlerin azaltılmasını amaçlamakta fakat tam tersine ekstra maliyetler ortaya çıkarmaktadır. Yapılan araştırmalara göre görüşme süresi sınırlaması yapmayan şirketlerde, telefon görüşmelerinin ortalama %15'i 3 dakikadan uzun sürmektedir. Görüşme süresini sınırlamış şirketlerde kesilen görüşmelerin %13,7'si ile kesilmenin hemen ardından tekrar görüşme başlatılmakta ve görüşmenin toplam süresi uzatılmaktadır. Bu uygulama, maliyetleri gerçek anlamda düşürmediği gibi iş performansını da olumsuz etkilemektedir. Şirketin iş amaçlı görüşmelerinde gerçekleştirilen kesme işlemi, önemli iş diyaloglarının yerine getirilememesine neden olmaktadır.

Tasarlanan sistem, sesli iletişim denetiminde genel çözümler yerine sonuç odaklı çözümler ortaya koymaktadır. Uzun görüşmelerin niteliğine bakılmadan engellenmesi yerine denetlenmesi ve belirli koşullar altında izin verilmesini mümkün kılmaktadır. Bu sayede iş süreçleri açısından gerekli olabilecek görüşmelerin kesintiye uğraması engellenmekte, kullanıcıların kurulu kuralları aşacak uygulamalara gitmesinin önüne geçilmektedir.

Başta hizmet sektöründeki şirketler olmak üzere tüm şirketler, günlük iş süreçlerinde telefon görüşmelerine yer vermektedir. Telefon kullanımına getirilecek sınırlamalar, şirketin iş geliştirme potansiyelini doğrudan etkilemektedir. Yanlış görüşme disiplini politikaları bu potansiyeli arttırmak yerine azaltmaktadır. Doğru görüşme disiplini uygulamaları ise şirketin iş potansiyelin arttırmakta, rakiplerine karşı rekabet gücünü yükseltmektedir. Sınırlandırmalar yüzünden gerekli görüşmeleri yapamayan şirketler iş fırsatlarını kaçırabilmekte bu da uzun vadede daha ciddi sorunlara neden olabilmektedir.

Uygulanacak görüşme disiplini politikası çalışanların aşırı telefon kullanımını engellememesi, otokontrol sağlamayı amaçlaması fakat bunları yaparken şirketin normal telefonla iş geliştirme süreçlerini aksatmaması gerekmektedir. Tasarlanan ve gerçeklenen sistem bu amaçlar doğrultusunda gerekli tanımlamaların yapılmasına, kuralların konulmasına ve işletilmesine imkan tanımaktadır.

7.2 Ön Ödemeli Servisler İle Katma Değer Uygulamaları

Bir kurumdaki müşterinin kalite gereksinimlerini ve beklentilerini karşılayacak ek özellik ve fonksiyonlar katma değer uygulamaları olarak adlandırılmaktadır. Bu tez çalışması ile gerçekleştirilen sistemin mevcut sesli haberleşme sistemlerine getirdiği özellik ve fonksiyonlar, kurumların telefon hizmetlerine katma değer sağlayacak uygulamalar olarak kullanılabilmektedir.

Tasarlanan sistem ile ortaya konulabilecek katma değerli uygulamalardan biri ön ödemeli sesli görüşme servisidir. Bu uygulama, okul, yurt, otel, hastane gibi ortak kullanıma açık telefonların yaygın kullanıldığı ortamlarda faturalı veya ankesörlü telefonların alternatif ön ödemeli telefon servisleriyle değiştirilmesi şeklinde gerçekleştirilmektedir. Buna göre kullanıcılar hizmeti veren kurumdan alacakları kontörler ile görüşmelerini gerçekleştirir, sistem de kullanıcının görüşme limitini yaptıkları ödeme çerçevesinde belirler.

Bu uygulamanın amacı yurt, hastane, otel gibi dönem sonunda faturalandırma yapılan yerlerde bulunan ve telefon maliyetleri konusunda hassas telefon kullanıcılarına önceden belirledikleri oranda görüşme yapma imkanı tanımaktır. Benzer olarak, görüşme limitlerinin doğrudan kullanıcılar yerine başka otoriteler tarafından belirlenmemesi gereken durumlar için de ön ödemeli sistemlerin kullanılmasına imkan tanınmaktadır.

Sistemin getirdiği bir diğer katma değer ise, ön ödemeli servisin verildiği kuruma ek gelir sağlanmasıdır. Kurum, ön ödemeli ortak ve açık bir telefon servisi vererek görüşmenin iletileceği farklı alternatif operatörler vasıtasıyla verdiği telefon hizmetinden gelir sağlayabilmektedir. Satılan kontör adedi ve yapılan görüşmelerin arama niteliğine bağlı olarak yapılabilecek tanımlamalar ve yönlendirmeler sayesinde servisi veren kurum görüşmeleri daha ucuza malederek verilen telefon hizmetinden gelir sağlayabilmektedir.

7.3 Numara Taşınabilirliği Uygulamaları

Numara taşınabilirliği, bir telefon abonesinin telefon numarasını değiştirmeden coğrafi

bölgesini, operatörünü veya tarifesini değiştirebilmesidir. Numara taşınabilirliğinin telefon kullanıcılarına ve servis sağlayıcılara sağladığı başlıca faydalar şunlardır:

- Servis aldığı operatörü değiştirmek isteyen kullanıcılar aynı numarayı kullanmaya devam edebilir.
- Kullanıcı kendisine daha cazip bir tarife planına geçiş yapabilir.
- Kullanıcı kendisine daha uygun gelen servis şartlarını sunan operatöre geçiş yapabilir
- Operatör değiştiren kullanıcı kendisini arayacak kişilere yeni bir numara bildirmek zorunda kalmaz.
- Operatörler kendilerine tahsis edilmiş numara havuzunu daha etkin kullanabilir.
- Operatörler numarasını değiştirmek istemeyen kullanıcıları da transfer edebilecek rekabet şartları ortaya koyabilir.

Özellikle Avrupa Birliği'nin ortaya koyduğu şartlar ve artan rekabet seviyesi, operatörlerin numara taşınabilirliği konusunun ele alınmasında etkili olmuştur. Çizelge 7.1'de Avrupa'daki mobil numara taşınabilirlik uygulamalarının başlangıç tarihleri ve 2006 yılına kadar taşınan numara sayıları verilmektedir.

Çizelge 7.1 Avrupa'da mobil telefon taşınabilirliği başlangıç tarihleri ve 2006 itibarıyla sayıları

Ülke	**Uygulama Başlangıcı**	**Taşınan Numara Sayısı**
İngiltere	1999	3.036.863
Hollanda	1999	925.343
İspanya	2000	1.210.000
Danimarka	2001	918.000
İsveç	2001	486.936
Portekiz	2002	33.400
İtalya	2002	1.600.000
Fransa	2003	100.000
İrlanda	2003	142.000

Numara taşınabilirliğinin başlatıldığı ülkelerde mobil telefon kullanıcılarının ortalama %10'unun operatör değiştirdiği görülmüştür. Bu oran, operatörlerin verdiği servislerin temelde benzer seviyede olduğu ve geçişlerin temel nedenlerinin tarifeler ve telefon

maliyetleri olduğu ülkelerde %15'lere vardığı hesaplanmıştır. Buna göre yakın bir dönemde Türkiye'de gerçekleşecek uygulamada da, mobil telefon kullanıcılarının %15'inin operatör değiştireceği düşünülmektedir.

Numara taşınabilirliği kullanıcı ve operatörler açısından önemli faydalar getirmesine rağmen, özellikle tasarruf amacıyla FCT'lerin yaygın olarak kullanıldığı kurumlarda bir takım problemler ortaya çıkarmaktadır. Bu kurumlarda, dış yönde yapılan GSM aramalarında aranılan operatöre ait FCT kullanılarak, görüşme hesaplı tarifeden gerçekeştirilmektedir. Fakat numaranın başka bir operatöre taşındığı durumda, doğru FCT cihazı seçilemeyeceğinden hedeflenen tasarruf elde edilememektedir. Buna göre görüşmelerin %20'sinin GSM yönünde yapıldığı bir şirkette aranan numaraların %10'unun numara taşınabilirliğinden yararlandığı varsayıldığında, tüm görüşmelerin %2'sinin hesaplı olmayan bir tarifeden gerçekleştirileceği sonucuna varılabilir. Bir numarayı farklı operatörden aramanın maliyeti, aynı operatörden aramanın 4 katı kadar olduğu düşünülürse telefon giderleri üzerinde %10'a varan bir ek maliyetin oluşması kaçınılmazdır. Artan GSM görüşme oranıyla ortaya çıkacak kayıp da artmaktadır.

Bu tez çalışması sonuıcu tasarlanan sistem, numaranın operatöründen bağımsız olarak hangi FCT'den aranması gerektiğini belirleyebilmekte, böylece numaranın taşınması durumunda dahi en hesaplı FCT üzerinden görüşmenin gerçekleştirilmesi sağlanmaktadır. Taşınan numaraların tanımı arayüz üzerinden gerçekleştirilebileceği gibi, mevcut bir CDR sistemi ile entegrasyon sağlanarak tanımlamaların otomatik olarak gerçekleştirilmesi de sağlanabilmektedir. Yine operatörlerin sunduğu elektronik fatura ve raporlama sistemleriyle entegrasyon da sağlanarak bu işlemlerin ekstra bir kullanıcı girdisi gerektirmeden otomatik ve şeffaf olarak gerçekleştirilmesi de sağlanabilecektir.

7.4 Çağrı Yönlendirme Temelli Verimlilik Uygulamaları

İş süreçlerinde telefonu yoğun olarak kullanan şirketlerde telefon süreçlerinin doğru planlanması verimlilik uygulamaları açısından çok büyük önem arzetmektedir. Yoğun telefon trafiği olan ortamlarda, özellikle yanlış kişilere bağlanan telefonlar iş performansını önemli oranda düşürebilmektedir. Bu nedenle, telefon trafiğini düzenleyebilen bir sistem, önemli oranda iş faydası üretmektedir.

Kurumun yapısına ve faaliyetlerine bağlı olarak tez çalışması sonucu tasarlanan ve gerçeklenen sistem fonksiyonları ve özellikleri çerçevesinde çağrı yönlendirme temelli bir çok

uygulama tasarlanabilmektedir. Örneğin satış faaliyetleri yürüten bir şirkete yapılan aramalarda kredisi dolan veya ödemelerini geciktiren bir müşteri satış bölümü yerine doğrudan tahsilat bölümüne bağlanabilir. Bu uygulama hem satış bölümünün iş zamanına katkıda bulunurken hem de şirketin tahsilat sürecini hızlandırmaktadır.

Geliştirilebilecek başka bir uygulamada, destek faaliyetleri yürüten bir şirketin anlaşmalı veya anlaşmalı olmayan taraflarca yapılan aramaları farklı departmanlarla karşılaması sağlanabilir. Bu uygulama, destek personelini her seferinde destek kapsamını belirleme işinden kurtararak iş sürecinde verimlilik sağlamaktadır.

Telefon süreçlerinde verimlilik sağlayabilecek bir diğer uygulamada, müşteri temsilciliği yapısında faaliyet gösteren bir firmada, arayan müşterinin kendi temsilcisiyle görüşmesi otomatik olarak gerçekleştirilebilmektedir. Benzer olarak arayanlara sabit bir temsilcinin atanmadığı durumlarda arayan kişinin son görüştüğü kişiye otomatik olarak bağlanması sağlanabilmektedir. Bu sayede arayan kişi, her seferinde farklı kişilere önceki görüşmelerde konuşulan konuları tekrarlamak zorunda bırakılmamış ve dolayısıyla görüşme sürelerinde tasarruf elde edilmiş olmaktadır. Bu uygulamayla telefon bağlama sürelerinin kısaltılması ve bunun sonucu gerçekleşen yüksek müşteri memnuniyetiyle artan fayda ve verimlilik sağlanabilmektedir.

Günümüz telefon sistemleri tarafından yerine getirilen tek yönlendirme uygulaması, aranan kişinin durumuna veya konumuna göre çağrının yönlendirilmesidir. Bu uygulamada, kullanıcılar gezi, toplantı vb. nedenlerle ofisleri dışında olduklarında ofise gelen çağrının telefon sistemi tarafından otomatik olarak bulunulan mekana veya mobil telefona yönlendirmesi sağlanmaktadır. Benzer şekilde yönlendirme servisi, çağrıyı arayan kişinin profiline göre kullanıcının ofis telefonuna, mobil telefonuna veya ev telefonuna yönlendirebilmektedir. Bu uygulama kullanıcıya iletişim olanaklarından vazgeçmeden hareket imkanı kazandırmakta ek olarak aramalar üzerinde seçici erişim sağlamaktadır. Fakat bu uygulamayı yerine getiren sistemler, gerektirdikleri lisans şartları ile yüksek maliyetler oluşturmakta, sadece büyük ölçekteki firmalar tarafından tercih edilmektedir.

Bu tez çalışması sonucu tasarlanan ve gerçeklenen sistem, yukarıda örneklendirilen uygulamalar da dahil olmak üzere farklı yapıdaki kurumlara uygun yeni ve dinamik verimlilik uygulamaları ortaya koyabilmektedir. Tüm bu uygulamalar kullanılan telefon sistemi ve altyapısından bağımsız olarak gerçekleştirilebildiğinden her ölçekteki kurumlarda gerçekleştirilebilmekte, hedeflenen verimlilik düzeylerine ulaşmak üzere yeni telefon

süreçleri tanımlanabilmektedir.

7.5 Spam çağrı engellemeye dayalı verimlilik uygulamaları

Telefon süreçlerinin iş süreçleri içinde önemli yer kapladığı kurumlarda, istenmeyen telefon çağrılarının önlenmesi, iş verimliliği açısından büyük önem arzetmektedir. Çalışanların istenmeyen çağrılarla meşgul edilmemesi, iş sürelerine katkıda bulunacağından kurumun iş verimine de katkıda bulunmaktadır.

Özellikle çağrı merkezleri başta olmak üzere yoğun olarak telefon karşılayan sistemler, sıklıkla amaç dışı çağrıların hedefi olmakta, bu çağrılar nedeniyle verilen servisin cevap süreleri artmakta ve dolayısıyla hizmet kalitesi düşmektedir. Bu yüzden çağrıların niteliklerinin belirlenmesi ve filtrelenmesi önemli bir ihtiyaç olmaktadır.

Bu tez çalışması sonunda tasarlanan ve gerçeklenen sistem, bir kurumun telefon sisteminde yapılan tanımlamalar çerçevesinde istenmeyen görüşmelerin engellenmesini sağlayabilmektedir. Bu uygulama, kurum çapındaki tüm kullanıcıların veya bir bölümünün belirli bir numara ile görüşmesinin engellenmesi şeklinde olabileceği gibi, belirli bir arama deseni oluşturan çağrıların sistem tarafından izlenmesi, belirlenmesi ve istenilen süreler çerçevesinde engellenmesi şeklinde de gerçekleştirilebilmektedir.

Bu özelliklere sahip bir uygulama, belirli bir zaman dilimi içinde sürekli arayan numaraların engellenmesi şeklinde gerçekleştirilebilir. Örneğin, bir kullanıcı, belirli bir numara tarafından sürekli aranarak meşgul ediliyorsa, bu numaranın bir süre için sistem içinde bağlanması engellenebilir. Böylece kullanıcılar bu numara tarafından meşgul edilmesi önlenebilir.

Bir diğer uygulama, kurumun belirlediği bir numara grubunun kurumdaki telefon kullanıcıları tarafından aranamaması şeklinde olabilmektedir. Örneğin kurum yetkilileri, rakip firmaların veya iş süreçleri ile ilgisi olmayan telefon servislerinin aranmasını engelleyerek bir telefon politikası oluşturabilir.

Internet telefonunun yaygınlaşmasıyla artan telefonla pazarlama faaliyetleri, yakın bir zamanda numaraların filtrelenmesi ihtiyacının daha da belirginleşmesine neden olacaktır. Internet telefonunun yapısı nedeniyle bu faaliyetlerin engellenmesi işi ulusal regülasyonlardan çok telefon sistemlerine düşecektir. Tez çalışması sonucu tasarlanan sistem, bu faaliyetleri filtreleyebilmekte ve kullanıcıları bu faaliyetlerin zararlı sonuçlarından korumaktadır.

8. SONUÇ

Günümüz sesli iletişim teknolojilerindeki gelişmeler ve iletişim sektöründe yapılan düzenlemeler sonucu artan telefon kullanımı, sesli iletişim alanında yeni ihtiyaçların ortaya çıkmasına neden olmuş, iş süreçlerininin etkinliğinin arttırılması ve sesli iletişim maliyetlerinin azaltılması gibi amaçlarla sesli iletişim trafiğinin izlenmesi ve kontrol edilmesi mevcut sesli iletişim sistemlerinden beklenen önemli bir fonksiyon haline gelmiştir. Gerçekleştirilen tez çalışmasında bu fonksiyonu karşlımaya yönelik bir mimari oluşturulmuştur. Bu mimari, JAVA programlama dili ve Asterisk PBX kullanılarak gerçeklenmiştir. Gerçekleştirilen uygulama ile bir kurumdaki iletişim trafiğine yönelik senaryolar belirlenmiş ve bu senaryolar uyarınca yapılan testler sonucu aşağıdaki sonuçlara varılmıştır.

1) Sistem CTI teknolojisi kullanılarak gerçeklenmiştir. Bu yapı, farklı teknoloji ve altyapıya sahip telefon sistemleri için ortak, geniş fonksiyonaliteye sahip, düşük maliyetli ve yüksek performanslı bir çözüm ortaya koymaktadır.

2) Sistem, modüler yapıda tasarlanmış ve gerçeklenmiştir. Sistemin kaynaklara ulaşım ve mantıksal kısımları paralel çalışabilecek yapıda oluşturulmuştur. Bu özelliği sayesinde sistem istenilen oranda ölçeklendirilebilmekte ve yüksek performans sağlamaktadır.

3) Sistem, kullanıcılar arasındaki sesli iletişimi aksatacak seviyede bir gecikme oluşturmayacak yapıda tasarlanmış ve gerçeklenmiştir. Sistemin gerçekleştireceği denetim ve işlemler dolayısıyla oluşabilecek gecikmeler, paralel çalışabilen modüller sayesinde sesli haberleşme sistemlerinin kabul edilebilir seviyelerinin altında olması sağlanmıştır.

4) Gerçeklenen sistem kurumlarda telefon kullanıcılarının iş amaçlı olmayan görüşmelerini sınırlandırabilmektedir. Bu sınırlandırma sonucu telefon maliyetlerinde %40'a varan tasarruf elde edilebilmektedir.

5) Sistem, iş amaçlı olmayan görüşmeleri sınırlayarak bu sürenin iş amaçlı süreçlere kazandırılabilmektedir. Denetim yapılmadığında toplam telefonla görüşme süresinin %30'una kadar varabilen bu zaman dilimi geri kazandırılarak kurumun verimliliğine katkı sağlanmaktadır.

6) Görüşmelerin kontrolü ve sınırlandırılması, mevcut uygulamaların aksine çalışan motivasyonunu ve iş süreçlerini olumsuz etkilemeyecek şekilde gerçekleştirilmektedir. Bu çerçevede tüm işlemler belirlenen kurallarla tanımlanan çağrılar için uygulanmaktadır.

7) Sistem, gerçekleştirilen tüm çağrılar üzerinde tam kontrole sahip olduğundan kurumların mevcut sesli iletişim sistemlerine katma değerli yeni hizmetleri ekleyebilmelerine imkan tanımaktadır. Tasarlanan sistem, kullanılan sesli iletişim sistemi ile entegre ve dinamik bir yapı sunarak kurumun özel ihtiyaçları doğrultusunda mevcut sistemin desteklemediği yeni fonksiyonların eklenebilmesini sağlamaktadır.

8) Sistem, numara taşınabilirliği uygulamaları sonucu ortaya çıkan ek maliyetlerin önlenmesi işlemini yerine getirebilmektedir. Bunun sonucu olarak FCT'lerin yoğun olarak kullanıldığı kurumlarda %10 oranında oluşabilecek ek maliyet önlenebilmektedir.

9) Tasarlanan sistem, gerçekleştirilen numara yönlendirme uygulamalarıyla kurum verimliliğine ve müşteri memnuniyetine önemli katkılar sağlayabilmektedir. İş süreçlerindeki ve kullanıcı durumlarındaki değişimlere yönelik tanımlanabilen kurallarla kurum içindeki telefon trafiğinin kontrolü sağlanmakta, görüşmelerin hedeflenen şekilde gerçekleştirilerek daha etkin hale getirilmesi mümkün olmaktadır.

10) Gerçeklenen sistem, kurumun telefon ve dolayısıyla iş süreçlerini etkileyecek çağrıların gerçekleştirilmesini engelleyebilmektedir. Tanımlanan kurallar çerçevesinde kurumun kaynaklarını olumsuz etkileyebilecek aramalar engellenerek iş verimine katkı sağlanmaktadır.

Yapılan testler ve sistemin çalışması göz önüne alınarak, sistem üzerinde ileriye yönelik geliştirmeler yapılabileceği sonucuna varılmıştır. Sistemin farklı uygulamalarla iletişim kurması sağlanarak karar mekanizmasına yeni yetenekler kazandırılabilmektedir. Farklı uygulamalarda kullanılabilecek PBX'lerin desteklediği ek fonksiyonlara yönelik yeni komutlar sistemin karar mekanizması tarafından desteklenmek üzere eklenebilmektedir. Mevcut sistem, değişik üçüncü şahıs uygulamalar ve PBX'ler ile entegre olabilecek yapıda tasarlanmıştır.

KAYNAKLAR

MacIntosh R., Vinokurov D, 2005, “Detection and Mitigation of Spam in IP Telephony Networks using Signaling Protocol Analysis”, Advances in Wired and Wireless Communication, 2005 IEEE/Sarnoff Symposium, 2005.

Turner K. J., Reiff-Marganiec S., Blair L., Pang J., Gray T., Perry P., Ireland J., 2005, “Policy support for call control”, Computer Standards and Interfaces. v28 i6. 635-649.

Jevtic D., 1998, “Unsuccessful Calls in PBX – Human Factor Considerations”, Electrotechnical Conference, MELECON 98., 9th Mediterranean, 1998.

Wu X., Schulzrinne H., 2005, “Service Learning and Service Risk Management in Internet Telephony”, IEEE International Conference on Communications (ICC'05), Seoul, Korea, 16-20 May 2005.

Erlang A.K., 1909, "The Theory of Probabilities and Telephone Conversations", Nyt Tidsskrift for Matematik B, vol 20, 1909

Erlang A.K., 1917, "Solution of some Problems in the Theory of Probabilities of Significance in Automatic Telephone Exchanges", Elektrotkeknikeren, vol 13, 1917.

INTERNET KAYNAKLARI

[1] “SPIT (VoIP Spam)”, http://en.wikipedia.org/wiki/Spit_%28VoIP_spam%29

[2] “Java Teknolojisi ile İlgili Bilgi Edinin”, http://www.java.com/tr/about/

[3] “What is Asterisk?”, http://asterisk.org/about

[4] “Learn About Java Technology”, http://www.java.com/en/about/

[5] http://www.cisco.com/

[6] http://www.alcatel-lucent.com

[7] http://www.karel.com.tr/

[8] http://en.wikipedia.org/wiki/TCP_IP

[9] http://www.itu.int/rec/T-REC-E.164/en

[10] http://www.itu.int/rec/T-REC-E.123/en

[11] http://www.linux.org/

[12] http://www.apple.com/macosx/

[13] http://www.openbsd.org/

[14] http://www.freebsd.org/

[15] http://www.sun.com/software/solaris/

[16] http://www.ecma-international.org/activities/Communications/TG11/cstaIII.htm

[17] http://www.att.com

[18] http://www.novell.com/

[19] http://www.webopedia.com/TERM/T/TSAPI.html

[20] http://www.sun.com/

[21] http://java.sun.com/products/jtapi/

[22] http://www.microsoft.com

[23] http://msdn2.microsoft.com/en-us/library/ms811413.aspx

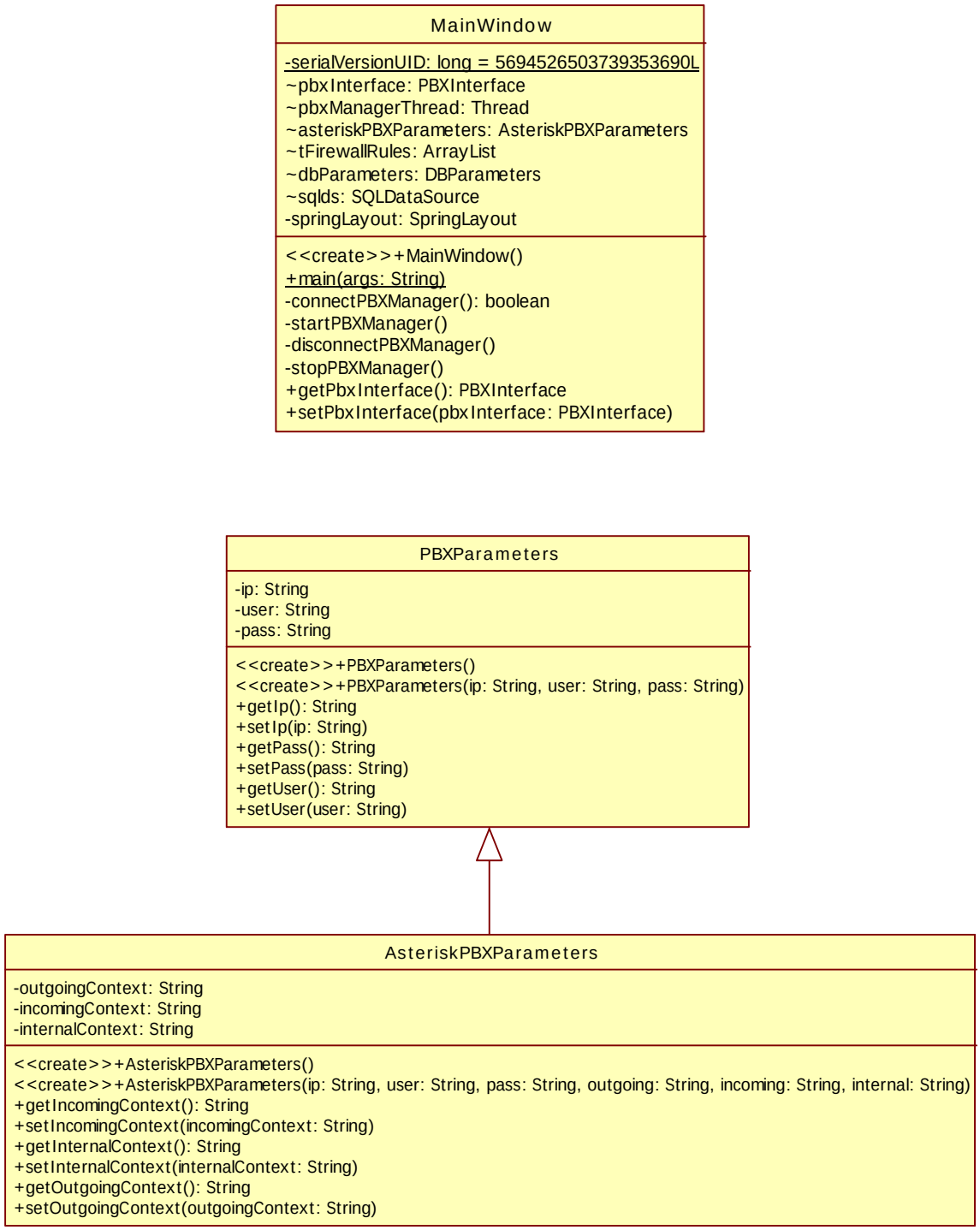
[24] http://jtds.sourceforge.net/

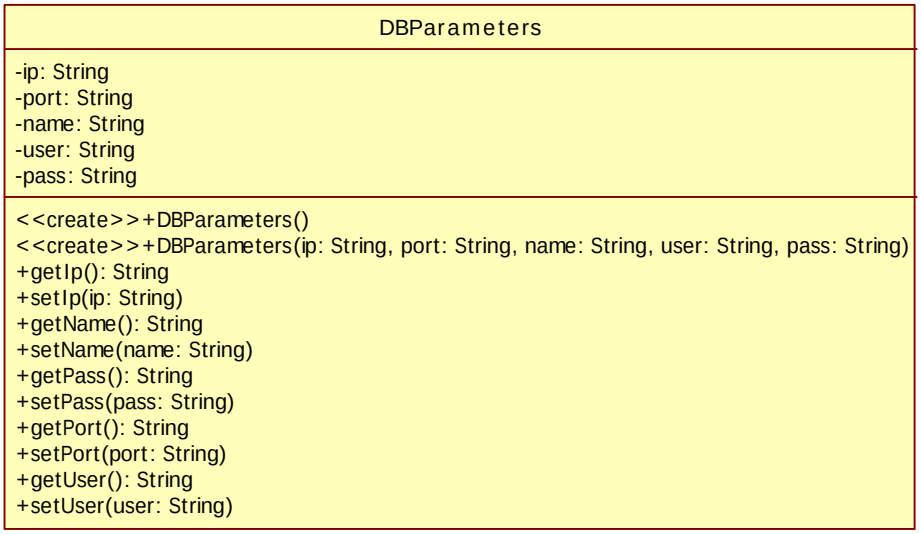
[25] http://www.ipera.com.tr

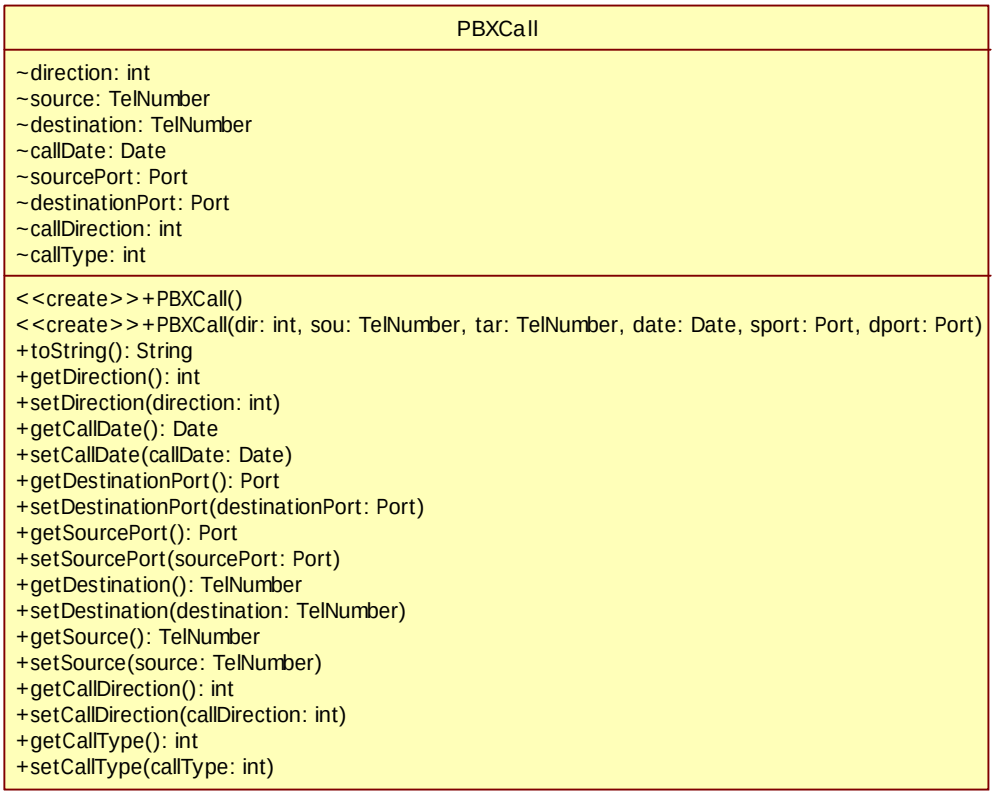
[26] http://www.ipera.com.tr/icerik/hizmetler/gorusmedisiplini/main.html

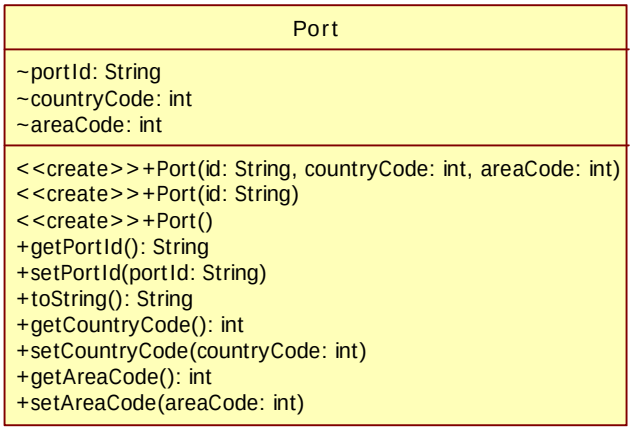
EKLER

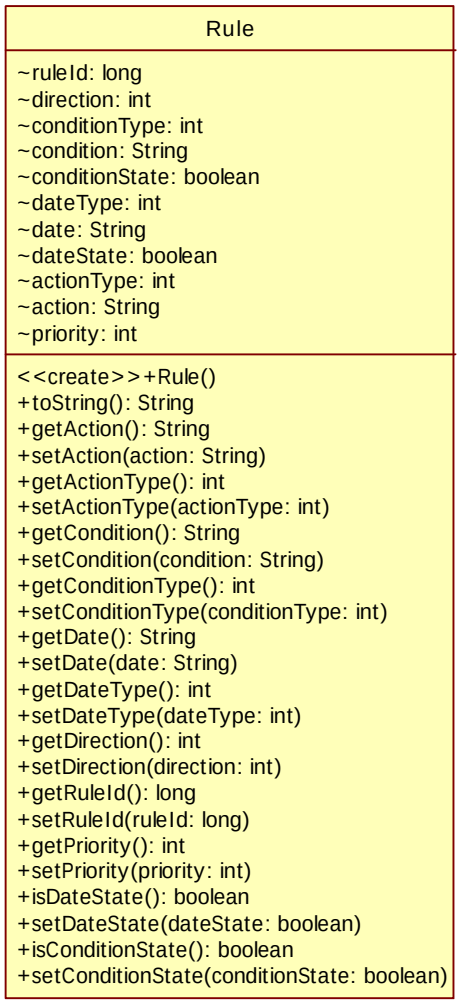
Ek 1 Uygulamanın JAVA sınıf diyagramları

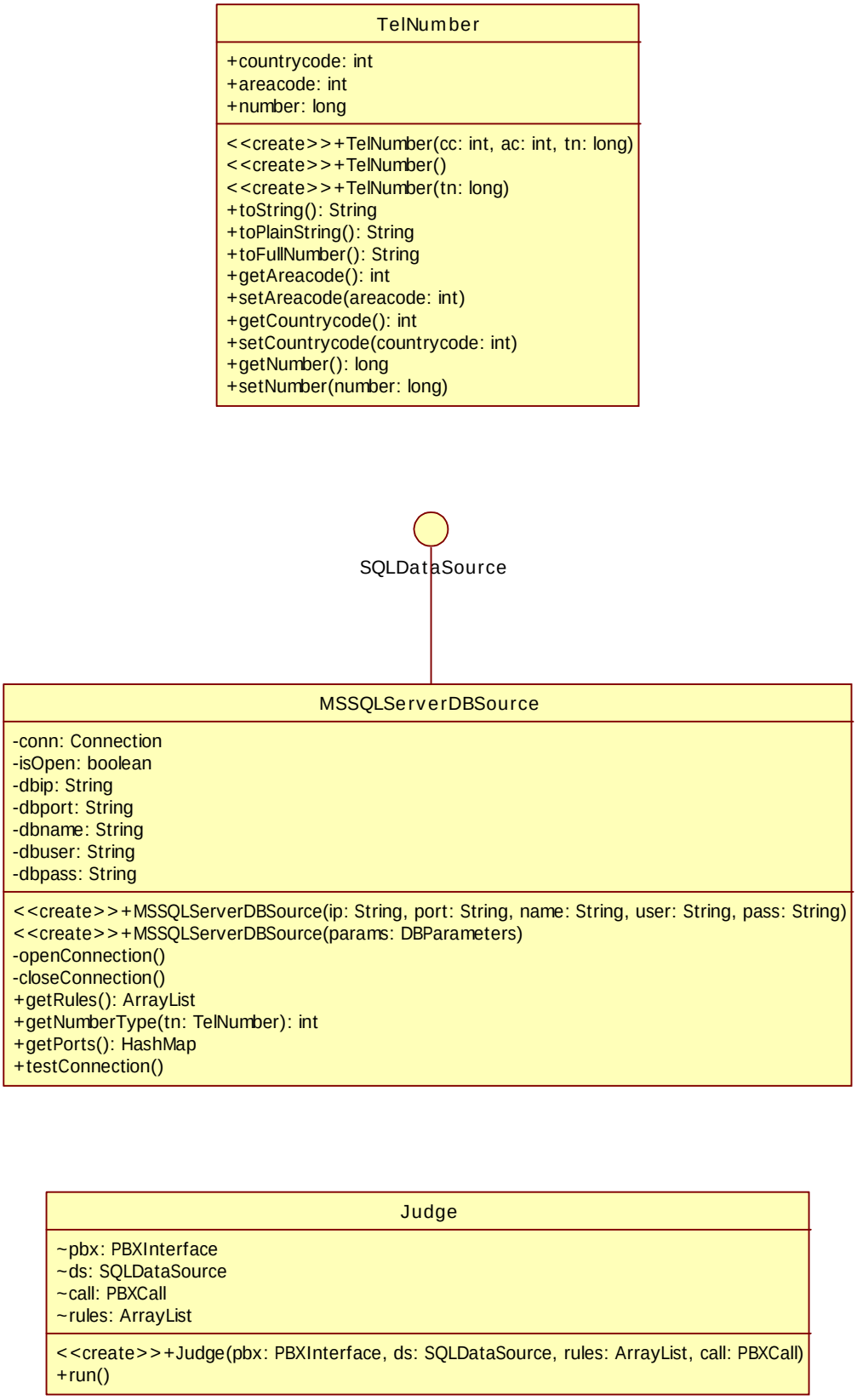


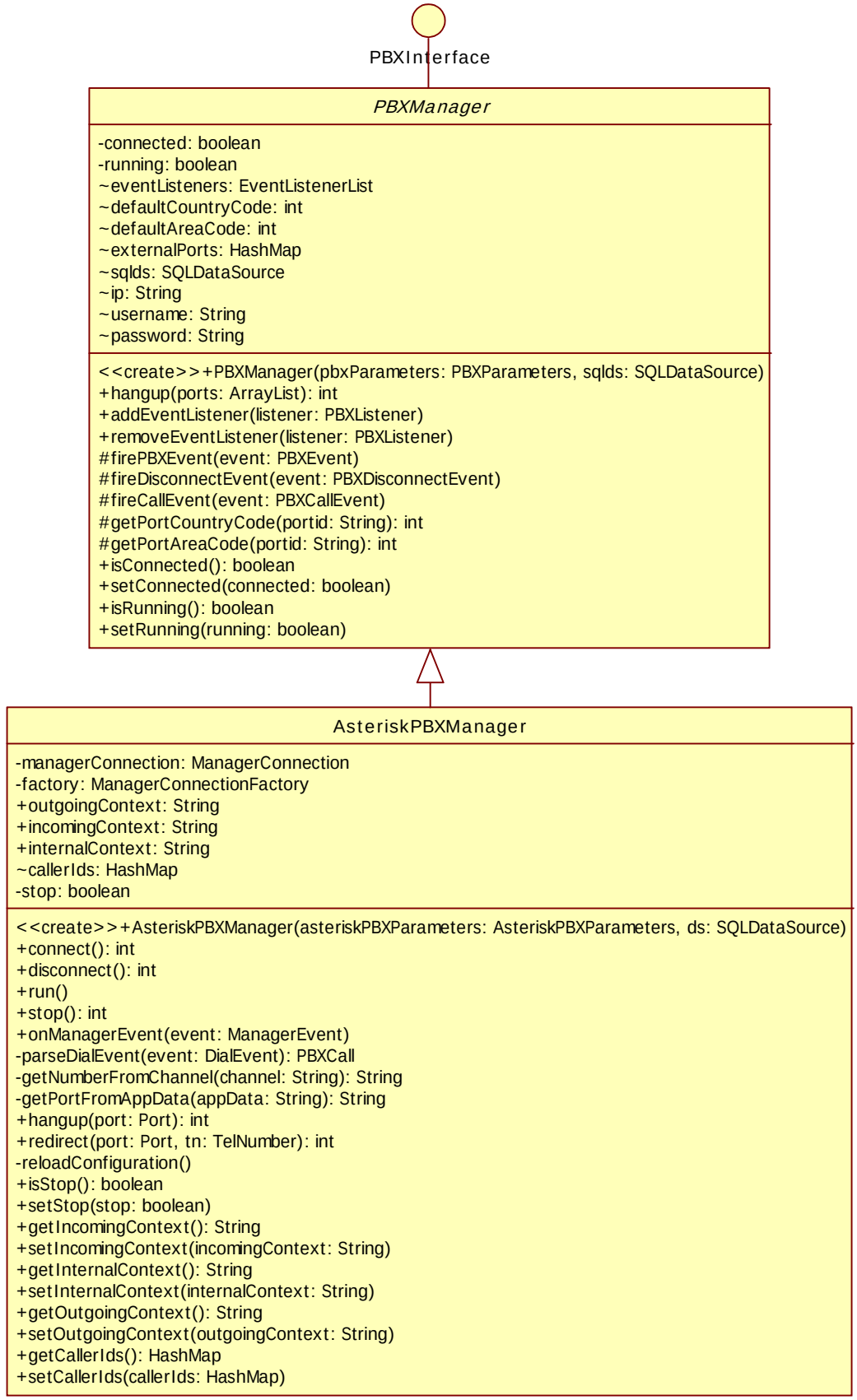

DBParameters
-ip: String
-port: String
-name: String
-user: String
-pass: String
<<create>>+DBParameters()
<<create>>+DBParameters(ip: String, port: String, name: String, user: String, pass: String)
+getIp(): String
+setIp(ip: String)
+getName(): String
+setName(name: String)
+getPass(): String
+setPass(pass: String)
+getPort(): String
+setPort(port: String)
+getUser(): String
+setUser(user: String)

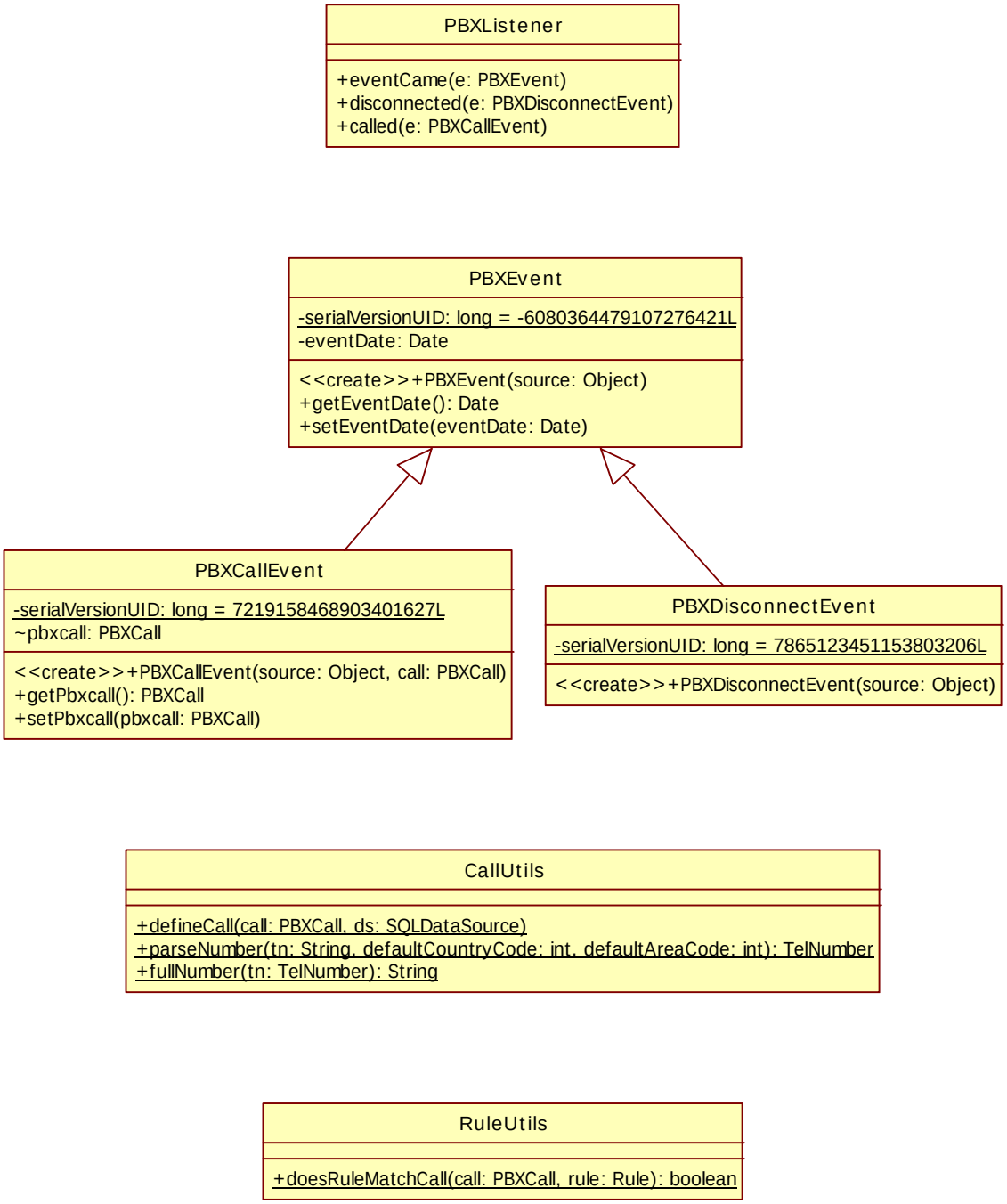

PBXCall
~direction: int
~source: TelNumber
~destination: TelNumber
~callDate: Date
~sourcePort: Port
~destinationPort: Port
~callDirection: int
~callType: int
<<create>>+PBXCall()
<<create>>+PBXCall(dir: int, sou: TelNumber, tar: TelNumber, date: Date, sport: Port, dport: Port)
+toString(): String
+getDirection(): int
+setDirection(direction: int)
+getCallDate(): Date
+setCallDate(callDate: Date)
+getDestinationPort(): Port
+setDestinationPort(destinationPort: Port)
+getSourcePort(): Port
+setSourcePort(sourcePort: Port)
+getDestination(): TelNumber
+setDestination(destination: TelNumber)
+getSource(): TelNumber
+setSource(source: TelNumber)
+getCallDirection(): int
+setCallDirection(callDirection: int)
+getCallType(): int
+setCallType(callType: int)


Port
~portId: String
~countryCode: int
~areaCode: int
<<create>>+Port(id: String, countryCode: int, areaCode: int)
<<create>>+Port(id: String)
<<create>>+Port()
+getPortId(): String
+setPortId(portId: String)
+toString(): String
+getCountryCode(): int
+setCountryCode(countryCode: int)
+getAreaCode(): int
+setAreaCode(areaCode: int)


Rule
~ruleId: long
~direction: int
~conditionType: int
~condition: String
~conditionState: boolean
~dateType: int
~date: String
~dateState: boolean
~actionType: int
~action: String
~priority: int
<<create>>+Rule()
+toString(): String
+getAction(): String
+setAction(action: String)
+getActionType(): int
+setActionType(actionType: int)
+getCondition(): String
+setCondition(condition: String)
+getConditionType(): int
+setConditionType(conditionType: int)
+getDate(): String
+setDate(date: String)
+getDateType(): int
+setDateType(dateType: int)
+getDirection(): int
+setDirection(direction: int)
+getRuleId(): long
+setRuleId(ruleId: long)
+getPriority(): int
+setPriority(priority: int)
+isDateState(): boolean
+setDateState(dateState: boolean)
+isConditionState(): boolean
+setConditionState(conditionState: boolean)


TelNumber
+countrycode: int
+areacode: int
+number: long
<<create>>+TelNumber(cc: int, ac: int, tn: long)
<<create>>+TelNumber()
<<create>>+TelNumber(tn: long)
+toString(): String
+toPlainString(): String
+toFullNumber(): String
+getAreacode(): int
+setAreacode(areacode: int)
+getCountrycode(): int
+setCountrycode(countrycode: int)
+getNumber(): long
+setNumber(number: long)
SQLDataSource
MSSQLServerDBSource
-conn: Connection
-isOpen: boolean
-dbip: String
-dbport: String
-dbname: String
-dbuser: String
-dbpass: String
<<create>>+MSSQLServerDBSource(ip: String, port: String, name: String, user: String, pass: String)
<<create>>+MSSQLServerDBSource(params: DBParameters)
-openConnection()
-closeConnection()
+getRules(): ArrayList
+getNumberType(tn: TelNumber): int
+getPorts(): HashMap
+testConnection()
Judge
~pbx: PBXInterface
~ds: SQLDataSource
~call: PBXCall
~rules: ArrayList
<<create>>+Judge(pbx: PBXInterface, ds: SQLDataSource, rules: ArrayList, call: PBXCall)
+run()


PBXInterface
PBXManager
-connected: boolean
-running: boolean
~eventListeners: EventListenerList
~defaultCountryCode: int
~defaultAreaCode: int
~externalPorts: HashMap
~sqlds: SQLDataSource
~ip: String
~username: String
~password: String
<<create>>+PBXManager(pbxParameters: PBXParameters, sqlds: SQLDataSource)
+hangup(ports: ArrayList): int
+addEventListener(listener: PBXListener)
+removeEventListener(listener: PBXListener)
#firePBXEvent(event: PBXEvent)
#fireDisconnectEvent(event: PBXDisconnectEvent)
#fireCallEvent(event: PBXCallEvent)
#getPortCountryCode(portid: String): int
#getPortAreaCode(portid: String): int
+isConnected(): boolean
+setConnected(connected: boolean)
+isRunning(): boolean
+setRunning(running: boolean)
AsteriskPBXManager
-managerConnection: ManagerConnection
-factory: ManagerConnectionFactory
+outgoingContext: String
+incomingContext: String
+internalContext: String
~callerIds: HashMap
-stop: boolean
<<create>>+AsteriskPBXManager(asteriskPBXParameters: AsteriskPBXParameters, ds: SQLDataSource)
+connect(): int
+disconnect(): int
+run()
+stop(): int
+onManagerEvent(event: ManagerEvent)
-parseDialEvent(event: DialEvent): PBXCall
-getNumberFromChannel(channel: String): String
-getPortFromAppData(appData: String): String
+hangup(port: Port): int
+redirect(port: Port, tn: TelNumber): int
-reloadConfiguration()
+isStop(): boolean
+setStop(stop: boolean)
+getIncomingContext(): String
+setIncomingContext(incomingContext: String)
+getInternalContext(): String
+setInternalContext(internalContext: String)
+getOutgoingContext(): String
+setOutgoingContext(outgoingContext: String)
+getCallerIds(): HashMap
+setCallerIds(callerIds: HashMap)


PBXListener
+eventCame(e: PBXEvent)
+disconnected(e: PBXDisconnectEvent)
+called(e: PBXCallEvent)
PBXEvent
-serialVersionUID: long = -6080364479107276421L
-eventDate: Date
<<create>>+PBXEvent(source: Object)
+getEventDate(): Date
+setEventDate(eventDate: Date)
PBXCallEvent
-serialVersionUID: long = 7219158468903401627L
~pbxcall: PBXCall
<<create>>+PBXCallEvent(source: Object, call: PBXCall)
+getPbxcall(): PBXCall
+setPbxcall(pbxcall: PBXCall)
PBXDisconnectEvent
-serialVersionUID: long = 7865123451153803206L
<<create>>+PBXDisconnectEvent(source: Object)
CallUtils
+defineCall(call: PBXCall, ds: SQLDataSource)
+parseNumber(tn: String, defaultCountryCode: int, defaultAreaCode: int): TelNumber
+fullNumber(tn: TelNumber): String
RuleUtils
+doesRuleMatchCall(call: PBXCall, rule: Rule): boolean

ÖZGEÇMİŞ

Doğum tarihi	18.07.1981	
Doğum yeri	Ankara	
Lise	1992-1999	Beşiktaş Atatürk Anadolu Lisesi
Lisans	1999-2004	Yıldız Üniversitesi Elektrik Elektronik Fak. Bilgisayar Mühendisliği Bölümü
Yüksek Lisans	2004-2007	Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Müh. Anabilim Dalı, Bilgisayar Müh. Programı

Çalıştığı kurum(lar)

	2004-Devam ediyor	IPera İletişim Teknolojileri Bilgisayar Mühendisi