T.C.
YILDIZ TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BANKACILIK UYGULAMALARINDA İŞ ANALİZİ UYGULAMA TEKNİKLERİ

ÇİSEM ÖZKAN

YÜKSEK LİSANS TEZİ
MATEMATİK MÜHENDİSLİĞİ ANABİLİM DALI
MATEMATİK MÜHENDİSLİĞİ PROGRAMI

DANIŞMAN
PROF. DR. HÜLYA ŞAHİNTÜRK

İSTANBUL, 2019

İÇİNDEKİLER

Sayfa

BÖLÜM 3

BÖLÜM 4

SİMGE LİSTESİ

a_i	Hata değeri
c	Sınıf değeri
i	Başvuran kişi
m	Karar değişkeni
n_G	İyi kredi başvuru sonucu
n_B	Kötü kredi başvuru sonucu
W	Hataların toplamını minimize etmek için kullanılan ağırlık

KISALTMA LİSTESİ

BBE	Bireysel Borçluluk Endeksi
DFD	Data Flow Diagram (Veri Akış Diyagramı)
IIS	İnternet Bilgi Sunucusu
KKB	Kredi Kart Bürosu
KRM	Kredi Risk Merkezi
MVP	En Yalın Ürün Kavramı
OOP	Nesneye Yönelik Programlama
TBB	Türkiye Bankalar Birliği
TRIZ	Yaratıcı Problem Çözme Teorisi
UML	Birleşik Modelleme Dili
URL	Uniform Resource Locator (Standart Kaynak Bulucu)
VYŞ	Varlık Yönetim Şirketi
WEKA	Waikato Environment for Knowledge Analysis
YSA	Yapay Sinir Ağları

ŞEKİL LİSTESİ

Sayfa

ÇİZELGE LİSTESİ

Sayfa

ÖZET

BANKACILIK UYGULAMALARINDA İŞ ANALİZİ UYGULAMA TEKNİKLERİ

Çisem ÖZKAN

Matematik Mühendisliği Anabilim Dalı

Yüksek Lisans Tezi

Tez Danışmanı: Prof. Dr. Hülya ŞAHİNTÜRK

Kredi skorlama sistemi, bankaların ve diğer finansal kuruluşların kredi risklerini konsolide etmek ve müşterisine kredi vermek için doğru karara ulaşabilmek adına ihtiyaç duyulmaktadır. Finansal kuruluşların sahadaki kazanımı, hem zaman tasarrufu bakımından hem de ekonomik açıdan artı yönde ilerlemesi, kredi skorlama sistemlerinin geçirdiği aşamalar ve süreçlerden doğrudan etkilenmektedir.

Kredi başvurusu yapan müşteri için finansal kuruluşların aldıkları karar ve yöntemler, skor puan hesabı için kullanılan değerlendirme kriterlerine yön vermektedir. Bir fonksiyonun bağımsız değişkenlerine karşılık gelen kredi değerlendirme kriterleri üzerinde yapılan değişiklik, değişkenlerden oluşan fonksiyonun sonucunu etkilemekle birlikte bu sonuç karar destek topolojisine ışık tutmaktadır. Kredi değerlendirmede kullandığımız değişkenlerden bazıları; müşterinin sahip olduğu menkuller, yaş, meslek grubu, cinsiyet, müşterinin aylık net geliri, mesleği, öğrenim durumu,kredi kartlarının varlığı, düzensiz ödemeleri, kanuni takip durumu gibi kredi verirken dikkate alınan kredi değerlendirme kriterleridir. Kredi değerlendirme için kullanılan değişkenlerden bazıları ise sadece belli bir müşteri grubunu(özel sektör,emekli, firma çalışanı vb) etkilemektedir.

Müşteri grubu özelinde farklılaşan bu kriterler hem karar vericinin hem de müşterinin sistemden sağlayacağı faydayı ve sistemin de nasıl cevap vereceğini belirlemiş olacaktır.

Bu çalışmada mühendislik yaklaşımı olan modelleme üzerinden, sistemin nasıl çalıştığı ve nasıl davrandığının yol haritası çizilecektir. Çalışmanın amacı iş analizi teknikleri olarak analiz deseni, alan deseni ve tasarım desenleri üzerinden sistemin anlatılmasıdır. Kullanılan veri seti değişkeni, bireysel segmentli olan belirli bir müşteri tipi için uygulanacak ve kredi skoru hesabında kullanılan kriterlerin kredi sürecindeki rolünü belirleyecektir. Veri akışı içerisindeki etkisini modelleyebilmek adına analiz deseni yöntemlerine başvurulacaktır. Sistemin fonksiyonel spesifikasyon sürecine hazırlanması için, kredi skor puanlama sürecinde kullanılan modellemelerin oluşması önem teşkil etmektedir.

Uygulama kısmında ise; kredi skor puanı oluşumunda kullanılan değişkenlerin tanımının yapıldığı, revize edildiği, simüle edilerek rapor sonucunun izlendiği prototip tasarımının hazırlanarak arayüzlerin oluşturulduğu süreç ve analiz-tasarım deseni ile yazılan kodun eski kod ile karşılaştırarak performans ölçümü analiz edilecektir. Başarılı analiz/tasarım süreci ile ortaya konulan tüm çalışmalar en az hata ile kodlama yapılmasına imkan verecektir. Skorlama sisteminde kullanılan birçok modelleme bulunmakla birlikte bu çalışma kapsamında sistemin davranışını belirleyen modellemeler üzerinden sınırlandırılma yapılacaktır. Finansal bir kuruluş olan bankanın bireysel müşteri segmentine sahip müşteri datası üzerinden geçmiş dönem kredi başvurularına ait skor puanı hesaplanmasında kullanılan kriterler üzerinden yapılan değişikliklerin simüle edilmesi ve sonucunda kredinin red ya da kabul edilme kararının verilmesine yönelik skorlama sistemininin iş analizi teknikleri açısından geçirdiği aşamalar anlatılmaya çalışılmıştır. Karar vericinin doğru ve anlamlandırılmış bilgiye sadece önseziyle ulaşması yerine bankanın gerçek verisi üzerinden simüle edilmesi bankanın en az risk ile karar vermesinin yanında kendi iş kurallarının da iyileşmesine olanak vermiştir. Böylelikle finansal kuruluşların doğru karar vermesinin yanında kredi riskini tahminlemesi ile kar oranının olumlu yönde artışı sağlanacaktır.

Ele alınan bir diğer husus ise; bu analiz teknikleri ile yazılan kodun esnekliği, yeni isteklere hızlı adapte olması, sistemin modellenmeden önceki performans metrikleri ile sonraki performans metriklerinin ölçüm analizi(farklı senaryolarada response süre,istek sayısı ve data büyüklüğü karşılaştırması vb), yazılım kalitesini arttırmada model kullanımın önemi uygulamalar kullanılarak aktarılmaya çalışılacaktır.

Anahtar Kelimeler: İş Analitiği Teknikleri, Kredi Skoru, Analiz ve Tasarım, Performans İnceleme, Tasarım deseni

ABSTRACT

BUSINESS ANALYSIS APPLICATION TECHNIQUES IN BANKING APPLICATIONS

Çisem ÖZKAN

Department of Mathematics Engineering

MSc. Thesis

Adviser: Prof. Dr. Hülya ŞAHİNTÜRK

Banks and other financial institutions need credit scoring systems to coordinate credit risks and make the right decision to give credit to their customers. The processes and stages of credit scoring systems directly affect the success of financial institutions in their field both economically and in terms of time saving. The method followed by the financial institutions for their customers who apply for credit is the determination of the credit evaluation criteria which are effective in calculating the credit score score. Considering credit evaluation criteria as independent variables of a function; changes in variables affect decision output methodologies by influencing the output of the function generated by the variables. Some of the variables we call credit evaluation criteria are; It is the credit rating factors that are likely to be effective in decision making such as monthly net income, age, gender, occupation, educational status, possession of credit card / cards, legal follow-up status, irregular payments. However, some of these variables only affect a particular customer status-type (retired customer type, private sector customer type, etc.). With this distinction, it will make sense of how the customer and decision-making system will benefit from the system and how it will react.

In this study, the roadmap of how the system behaves and works will be presented through the engineering approach. For this purpose, it will be tried to explain the system through analysis pattern-area pattern and design patterns as business analysis techniques. In this study, the effect of the factors taken into account in the credit process will be shown when calculating the credit score for a particular customer type among the individual segmented customer data set variables. Analytical methods will be applied to model the data flow of this effect.

These models used in the process of credit score formation are important for preparing the system for the software process. In the last stage, the customers will be able to create the prototype design interfaces in which the variables used are defined, updated, simulated and the results of the reports are calculated, and the performance model will be analyzed by comparing the code written with the design model with the old code.

All studies will enable successful coding of a work that passes through a healthy analysis / design process. Although there will be many models used in this study, it will be limited to the main models that reveal the behavior of the system.

In this study, a credit scoring system applied for the decision of acceptance or rejection of credit as a result of this simulation is taken as a basis for calculating the score score of the previous period loan applications on a bank's individual customer segment data It was obtained. Simulating a modified data set in a real data set of a bank has enabled the bank to improve its core business metrics by making sense and risky decision making by accessing meaningful and accurate information. The results allow the financial decisions to make more accurate decisions and allow the profitability rates to increase with the estimation of the credit risk.

Study; the flexibility of code written with these analysis techniques, rapid adaptation to new requirements, performance analysis of performance metrics of the system prior to modeling and response analysis of performance metrics (response time, number of requests and data size comparison in different scenarios, etc.) will be tried to be conveyed by using applications to improve the quality of software.

Keywords: Business Analytics Techniques, Credit Score, Analysis and Design, Performance Review, Design Pattern

BÖLÜM 1

GİRİŞ

1.1 Literatür Özeti

Birçok sektörde problemi anlamak ve problem çözüm yöntemlerini belirlemek oldukça önemlidir. Hayatımızın her aşamasında ister iş hayatı ister günlük yaşantımız olsun karşılaştığımız problemleri modellemek ve onlara ait çözümleri tasarlayabilmek için analitik metodojileri kullanmak ve bilmek gerekmektedir. Hangi iş problemlerinin hangi teknikler ile çözülebileceğini öğrenmek her geçen gün yeni yaklaşımların doğmasını olanak sağlamaktadır. Analitik düşünmenin önemi ve veriye dayalı karar alma anlayışının anlatılması amacıyla günümüzde birçok teknik kullanılmakta ve büyük verinin rolü her geçen artmaktadır. Teknolojinin her geçen gün ilerlediği çağda problemin doğru şekilde ortaya konulması, eksiksiz verinin elde edilmesi ve karar vericinin işini kolaylaştırmak adına kullanılan yazılımın kalitesinin ölçülmesi, sürecin başarılı sonuçlanması adına iş analizi uygulama teknikleri kullanılarak analiz edilmektedir. Bu çalışma kapsamında bankacılık kredi skorlama sisteminde iş analizi uygulama teknikleri olarak analiz ve tasarım desenleri çevik yöntem ile uygulamalı olarak ele alındığında karar vericinin sistemden beklediği fayda simüle edilebilecek, sisteme dahil olması istenen herhangi bir geliştirme analiz deseni ile yazılan esnek kod sayesinde kolay sahaya alınabilecek ve desen kullanılarak, çevik yöntem ile yazılan kodun performansı waterfall yöntemdeki yazılan kod ile karşılaştırmalı olarak performans ölçümüne tabii tutulacaktır.

Literatürde; analiz ve tasarım deseni ayrı yada eş zamanlı incelenmiş, her iki desen de birçok araştırma konusunda kullanılarak sisteme sağladığı fayda ya istatistiksel ya da çeşitli uygulamalar kullanılarak ispat edilmiştir.

Analiz desenleri Fowler'a göre, iş modellemesinde ortak yapıyı temsil eden kavramlar grubudur[1]. Stoecklin ve Allen'a göre ise; tasarım desenleri yazılım gerçekleştirimleri için tekrar kullanılabilen yazılım yapılarıyken, analiz desenleri, yazılım ihtiyaçlarının ortaya çıkartılması ve temsil edilmesi için gerekli olan tekrar kullanılabilir soyutlamalar olarak ifade edilmektdir[2].

Bazı makalelerde, analiz desenlerinin zaman ve maliyet tasarrufu, daha yüksek kalite, daha az risk, standardizasyon, son kullanıcıyla verimli iletişim, tekrar kullanılabilirlik, alan detaylarını keşfetme, alanın anlatılması ve idamesinin kolaylaştırılması gibi yararları belirtilmiştir[3].Ancak diğer bazı makalelerde ise analiz desenlerinin tüm fonksiyonel ihtiyaçları karşılamadığı, gereksiz modellemelere neden olduğu, detay içermediği, isteğe göre uyarlama için çok zaman harcandığı, müşterilerin farklı gösterimleriyle yorumlandığında yanlış anlaşılmaların doğduğu gibi problemler yarattığı vurgulanmıştır[4].

Sesera "Temel Bankacılık Desenleri" makalesinde dar kapsamlı problemler için 24 bankacılık analiz deseni tanımlamıştır[5]. Sonrasında okuyucularının bu desenlerin nasıl bağlanılacağını ve bu desenlerin gerçek dünyadaki bankacılık sistemine uygun mu yoksa sadece ideal bankacılık sisteminin özelliklerini mi gösterdiğini sorması üzerine Sesera, literatürdeki kredi alanı için analiz desenlerini detaylıca belirten tek çalışma olan "Applying Fundamental Banking Patterns: Stories and Pattern Sequences" başlıklı makalesini yayımlamıştır[6].

Kazan, E çalışmasında analitik teknik olarak tasarım deseni kullanarak yazılan kodu önceki yazılan koda göre karşılaştırmalı olarak performans farkını analiz etmiştir. Performans incelemesi yapabilmek için, model-view-presenter, model-view-controler, ve proxy desenleri kullanılması ile geliştirilen kodun, aynı işlevdeki tasarım desenleri kullanılmayan koda göre karşılaştırması yapılmıştır. Yazılan her iki kod, aynı iş mantığına sahip kullanım senaryoları üzerinden test edilmiş, performans değerlendirme ölçümü yapılmış ve elde edilen sonuçlar analiz edilmiştir.

Çalışmada elde edilen sonuçlar, tasarım desenleri kullanılarak yazılan kodun işlemlere çok kısa sürede cevap verdiğini göstermiş ve süreçlerin tasarımında ilişkilerin çok iyi tanımlanmış olması yeni ihtiyaçları karşılama açısından kodun revize edilmesini kolaylaştırmıştır[7].

Al-Ahmad çalışmasında, endüstride ve akademide analitik teknik açısından ele alınan uygulamalarda tasarım desenlerinin kullanımını tanıtmaktadır. Çalışmasında tasarım desenlerinin otomatize edilerek daha detaylı tasarım modellerinin oluşumuna imkan veren tool önermektedir [8].

Ronald ve Nija çalışmalarında, tersine mühendislikte tasarım deseni kullanılması sonucunda yüksek performanslı yazılım mimarisi geliştirmeyi amaçlamışlardır. Ters mühendiste kullanılan araçların, performansı yüksek yazılımlarda kod tasarımlarının anlaşılması için önemli olduğunu savunmuştur. Durum ve strateji desenlerinde olduğu gibi bazı tasarım desenlerinin davranış açısından farklı, yapısal olarak aynı olduğunu ifade etmişlerdir. Tasarım desenlerindeki hata oranını minimuma indirmek için dinamik analizler uygulamışlardır. Ronald ve Nija' ya göre performansı yüksek yazılımlar tasarım deseni kullanılması ile yazılan kodlardır[9].

Woodside, Petriu ve Mani çalışmalarında, SOA yazılım modelinin tasarımındaki eksiklikler, mimarisindeki hatalar ve uygulamasında ortaya çıkan problemlerin çözülmesi için tasarım desenlerinin kullanılması gerektiğini belirtmişlerdir. Tasarımı kullanan kişi yazılım modellerini ve desenlerini tanımlamalarını UML kullanarak uygulamışlardır. Tasarımcı desenin uygulamasını kullanıcılara göre spesifikleştirerek tanımlamıştır. Aynı zamanda farklı çalışmalar üzerinde desenlerin etkilerini iterasyonel olarak yorumlamıştır[10].

Schmidt çalışmasında, yazılım mimari kalitesini artırmak ve sistem geliştirme aşamasında ortaya çıkan temel problemlerin önüne geçirilmesi adına tasarım deseninn kullanılmasını belirtmiştir. Bu problemler, tasarım paradigmalarının kullanımı, geliştiricinin hatalara düşmesi ve geliştiriciler arasındaki mimarideki bilgi paylaşımı, yeni mimari sitillerin kullanımı gibi konuları barındırmaktadır. Bu makale de, tasarım deseni kullanımı ile nesneye dayalı yazılımın arayüz iletişiminin yanında yeniden kullanıma olanak sağlaması hedeflenmiştir[11].

Doğan ve Koçyiğit çalışmalarında, nesne yönelimli analiz - tasarım bağlamında ve nesne yönelimli analiz-tasarım kabulleri ekseninde tekrar kullanılabilir, basit, uyarlanabilir, geliştirilebilir ve tasarıma ışık tutan kavramsal modeller ortaya koymayı ve bankacılık kredi alanı için analiz desenlerinin yararlarını göstermeyi hedeflemektedir. Bu amaçla, bir bankanın 13 yazılım gereksinim belirtim (SRS) dokümanı incelenmiş ve bankacılık kredi alanı için bir kavramsal model tanımlanmıştır. Bu tanımlanan desenin diger kredi veren organizasyonların iş alanı için de geçerli olduğunu doğrulamak için, 10 farklı organizasyonun kredi alanı iş ihtiyaçları ortaya çıkartılmış ve her organizasyonun alan modelleri ayrı ayrı tanımlanıp birbirleriyle karşılaştırılmıştır. Modeller arasındaki farklar göz önünde bulundurularak, ortak bir alan modeli elde edilmiş ve bu modelin geçerliligi, yararları ve yazılım tasarımıyla uyumu incelenmiştir[12].

Kalay ise çalışmasında, desen kullanılsın ya da kullanılmasın yazılan kodun kalitesini artırmak ve daha performanslı hale getirmek amacıyla; sınırlamalar yaparak statik kod analizi araçları kullanmış ve önemini vurgulamış, yazılım kalitesine sağladığı faydayı ve yazılım geliştirme sürecindeki katkısını ortaya koymuştur. Statik kod analizinin testlerde çıkabilecek hataların bir kısmını tespit etmesinin yanında; denetim ve atama gibi programlama problemlerin tespit ettiği ve sonuç olarak da yazılım testlerine işlevsellik sınanmasında bir seçenek olarak düşünülmemesi kanaatine varmıştır. Testlere başlamadan önce statik kod analizinin kaliteli yazılımları tamamlayıcı ve daha güvenilir olduğunu değerlendirmiştir. Elde ettiği bir diğer sonuç ise; statik kod analizi araçlarını birlikte kullanarak daha etkili sonuçlar elde ettiğini görmüştür [13].

Tez kapsamında veri seti olarak birçok araştırmada ortak alan deseninde kullanılan veri seti üzerinden çalışma yapılmıştır. Yıldız çalışmasında, Almanya'daki bir bankada "UCI Makine Öğrenim Deposu"nda bulunan xls formatında yer alan 1000 müşterisi üzerinden 20 farklı özelliği kullanılarak oluşturulmuş veri kümesini, bilgi keşfi sürecinden geçirmiştir. Verinin kullanılabilmesi için veri dönüştürme ve ön işleme tekniklerini WEKA yazılım ara yüzleri kullanılarak anlamlandırmıştır. Kullanılan veri seti üzerinden hesaplanacak olan kredi skorlamada en uygun sınıflandırma algoritmasını belirlemeye çalışmıştır.

Uygulamasında, veri madenciliği sürecindeki veri ön işleme işlemleri kullanımında veri setinde kullanılacak sınıflandırma algoritmaları için önem derecesi en az olanlar tespit edilerek veri setinden çıkarılması sağlanmıştır. Oluşturulacak hibrid sınıflandırma algoritmalarının, veri değerlendirme kriterleri(Doğruluk,F-Score,GiniKatsayısı, Duyarlılık,Kesinlik) ile analiz edilmiştir. Veri kümesinin veri madenciliği sürecine girmeden önce ve girdikten sonraki durumları karşılaştırılmış; hibrid sınıflandırma algoritmalarının diğer algoritmalara göre daha birbiri ile tutarlı sonuçların elde edildiğini gözlemlemiştir. Benzer şekilde bu tez kapsamında kredi skorlama sisteminde kullanılan veri seti için veri madenciliği teknikleri ile elde edilen küme üzerinden yorumlama yapılmıştır[14].

Donel çalışmasında [15] günümüzde kullanılan kredi skorlama modellerini belirtmiş ve ticari segmente ait müşteri veriseti üzerinden skorkartlar oluşturmaya çalışmıştır. Ele aldığı uygulama 2008 yılında yapılan tüketici kredisi başvurusu yapmış müşterilerin veri seti bilgilerini içermektedir. Skorkartların modellenmesi için yapay sinir ağları yöntemi ve lojistik regreyon yöntemini kullanmıştır. Birinci ve ikinci tip hata değeri üzerinden skorkartlardan elde edilen değerleri karşılaştırılmıştır. Birinci tip hata (iyi kredilerin kötü kredi olarak ifade edilmesi)tespitinde, lojistik regresyon modeli performansının daha iyi sonuç elde ettiği görülmüştür. Finansal kuruluşlar açısından daha maliyetli olan 2.tip hatanın (kötü kredilerin iyi kredi olarak ifade edilmesi) tespitinde ise; yapay sinir ağları modeli başarısı daha iyi sonuç elde etmiştir. Yapay sinir ağları toplam hata yüzdesinde daha düşük oranda kredi skorlama modeli oluşturduğu gözlemlenmiştir.

Soydaner çalışmasında, kredi skorlama modelleri oluşturulurken yapay sinir ağları(YSA), lojistik regresyon analizi, genetik algoritma(GA) ve lojistik regresyonu karşılaştırmalı olarak ele almıştır. Hibrit yapay sinir ağları, kredi skorlama verileri üzerindeki performansları bakımından klasik yapay sinir ağları ve lojistik regresyonla karşılaştırılmıştır. Yapılan analiz sonucunda, doğrusal olmayan ve esnek bir yapıya sahip olan yapay sinir ağlarının lojistik regresyon analizinden üstün olduğu ve ağ hibritleştirildiğinde daha da iyi sonuçlar elde edildiği görülmüştür [16].

Literatürde karşılaşılan kredi skorlama ile ilgili çalışmaların önemli bir kısmının aynı veri seti üzerinde yapılan çalışmalardan oluştuğu görülmüştür.

Bunun önemli bir nedeni bu konuda gerekli verinin kolay elde edilememesi, hatta ülkemiz dâhil pek çok yerde verinin paylaşımı konusunda yasal engellerin bulunmasıdır. Kullanılan veri setleri; UCI Machine Learning Repository veritabanında bulunan Alman kredi veri seti (German Credit) ve Avustralya kredi veri seti (Australian Credit) dir[17].

Alman ve Avustralya kredi veri seti üzerinde David West tarafından yapılan çalışmanın amacı, beş sinir ağı mimarisinin (radyal tabanlı fonksiyon, çok katmanlı algılayıcı, uzmanların karışımı, öğrenen vektör nicelemesi ve bulanık uyarlamalı rezonans) kredi skorlama için uygunluğunu araştırmak ve bunların performansını; Doğrusal Diskriminant Analizi, Lojistik Regresyon, k-enyakın Komşuluk, Kernel Yoğunluğu ve Karar Ağaçları yöntemleri ile karşılaştırmıştır. Bu araştırmanın sonuçları göstermiştir ki Sinir Ağı kredi skorlama modelleri, kredi skor doğruluğunda % 0.5 ten % 3'e kadar değişen ölçüde fraksiyonel olarak iyileştirme yapmıştır.

Ancak kredi skorlama modellerinde Sinir Ağlarının kullanımı, ağ topolojilerini geliştirmek ve üstün eğitim yöntemleri tasarlamak için biraz modelleme yeteneğine sahip olmayı gerektirir. Çok katmanlı algılayıcı (multilayer perceptron) en yaygın kullanılan sinir ağı modeli olmasına rağmen, uzmanların karışımı (mixture-of experts) ve radyal temelli fonksiyon (radial basis function) sinir ağları da kredi skorlama uygulamaları için düşünülmelidir. Bu araştırma, ayrıca Lojistik Regresyonun Sinir Ağları modellerine iyi bir alternatif olduğunu ileri sürmektedir. Birkaç Sinir Ağı eğitim yinelemesi içeren ortalama bir durum için Lojistik Regresyon, Sinir Ağı modellerinden biraz daha doğru sonuçlar üretir[16],[18].

Parametrik olmayan modeller (k-en yakın komşuluk ve Kernel Yoğunluğu) ve CART ümit verici sonuçlar üretmemiştir, ancak çok büyük veri setleri için iyileşmiş sonuçlar vereceği ispat edilebilir. Alman kredi veri setinde kredi verilen toplam 1.000 başvuru mevcut olup, bunların 700'ü iyi, 300'ü sorunlu kredidir. Her kayda ait kredi geçmişi, hesap bakiyesi, borç detayı ve kişisel bilgileri içeren 20 değişken mevcuttur. Avustralya kredi veri seti de buna benzer olup, 468'i iyi, 222'si sorunlu toplam 690 gözleme ait 14 değişkenden oluşmaktadır.

Bu iki verinin aşağıda aktarılacak pek çok farklı çalışmada kullanılmasının önemli bir yararı, bu çalışmaların sonuçlarının karşılaştırılmalarını mümkün kılmasıdır[14].

Sönmez çalışmasında, YSA metodolojisini, borçlunun temerrüde düşme ve kredi skorlama kararları gibi fnansal sorunların üzerine bir çözüm olarak sunmayı amaçlanmıştır. Finansal sorunlara çözüm getirmede özellikle YSA'ların genelleştirme özelliğinden faydalanılmaya çalışılmış, YSA metodolojisinin kullanıldığı yazılım modelinin sunduğu sonuçlar karar ağaçları metodolojisi ile geliştirilen modelden elde edilen sonuçlara göre tahminlemede daha yüksek başarı oranına sahip olduğunu ortaya koymuştur[19].

Kasapoğlu çalışmasında, kredi riski ölçümünde yeni modellerin anlatıldığı çalışmasında finansal kuruluşların kullandırdıkları kredilerin ödenmeme ihtimalini hesaplayabilmek adına lojistik regresyon modeliyle hazırlanan skorkart kullanmıştır. Bu skorkartı Gini Katsayısı, ROC eğrisi ve Kolmogorov-Smirnov istatistiği gibi yöntemler ile güvenilirliğini test etmiştir. Analizde kullanılan veri seti, 2007- 2008 Ocak tarihleri arasında ilgili finansal kuruluşa kredi başvurusunda bulunan müşterilerin başvuru formundaki bilgilerden, kredilerin iyi ya da kötü kredi olma karakteristiğinden ve müşterinin istihbarat bilgilerinden oluşmaktadır. Kredilerin karakteristiği regresyon modelinde bağlı değişken, müşteriye ait bilgiler ise bağımsız değişkenler olarak belirtilmiştir. Elde edilen sonuç, yeni başvuruların da geçmişte benzer performans sergilemiş müşterilerle benzer özelliklere sahip olmasının aynı performansı sergileyeceği düşüncesi ile, hazırlanan skorkartın finansal kuruluşun kredi verme performansını belirlemede etkili olduğunu belirtmiştir [20].

Turangil çalışmasında, bir bankanın kredi kartı kullanan müşterileri üzerinden kullanılan değişkenler ile iyi-kötü müşteri ayırımı yapılmış ve elde edilen sonuçlar birbirleriyle karşılaştırılmıştır. Uygulamada Lojistik Regresyon, Diskriminant Analizi, Kümeleme Analizi (Cluster Analysis), Sınıflandırma Ağaçları (Classification Tree) istatistiksel tekniklerin yanısıra; istatistiksel olmayan Yapay Sinir Ağları modeli kullanılmıştır. Çalışmada bu tekniklerin uygulamaları ele alınmış, teorik ayrıntılara değinmekten kaçınılmıştır. Uygulamada kullanılmayan sadece teorik açıdan bilgi amaçlı sunulan teknikler ise Tamsayı Programlama ve Lineer Programlama konularıdır.

Bu tekniklerle ilgili uygulamalar ileride yapılacak çalışmalara bırakılmıştır. Uygulamada kullanılan veri setine göre tahmin başarısı en yüksek olan modelin lojistik regresyon olduğu görülmüştür. Veri kümesine göre tüm modellerde kullanılan "aylık net gelir" değişkeni kredi skorlamada etkin değişken olarak gözlemlenmiştir [21].

Karakuş çalışmasında, Türkiye'de mobil telekomünikasyon sektöründe faaliyet gösteren bir firmaya ait belirli bir abone kitlesi için kredi risk sorunları incelemiştir. Davranışsal kredi skorlamada şüpheli alacaklı olarak düşünülen abonelerin önceden tahmin edilmesi ve zamanında önlem alınmasıyla elde edilecek zararın en aza indirgenmesi incelenmiştir. Bilgi keşfi uygulamaları kullanılarak geçmişte oluşan şüpheli alacakların yapısı analiz edilerek modellenmiştir. Ödeme performansını oluşturan değişkenlerin kullanıldığı veri kümesinde farklı veri madenciliği algoritmaları uygulanmıştır. Yapay Sinir Ağ, Lojistik Regresyon ve Karar Ağaçları modelleri söz konusu veri setine uygulanmıştır. Tutarlı ve birbirine yakın sonuçlar elde edilmesine rağmen Lojistik Regresyon modeli kullanılan modeller içerisinden en başarılısı olarak gözlemlenmiştir[22].

Tezde ele alınan çalışmanın farklı açıdan bir değerlendirmesi analiz-tasarım deseni üzerinden çevik yöntem esas alınarak yazılan kodun waterfall sürecinde yazılan koda göre performans incelemesinin yapılmasıdır. Performans incelemesi sonucunda restfull servis ile yazılan kodun istek çağrımları, server'ların cevap verme süreleri gibi kriterler esas alınarak sayısal açıdan etki süreleri ölçülmüştür.

1.2 Tezin Amacı

Örnek bir problemin ortaya konulması, doğru verinin elde edilerek tasarlanması, tasarım sürecinde yazılım metodojilerinin iş analitiği teknikleri ile anlatılması, karar vericinin doğru ve etkin karar vermesini sağlayacaktır.

Bu amaçla örnek problem olarak bankacılık sektöründe müşteriye kredi verme sürecinde doğru karar verilebilmesi için kredi skorlama sisteminin geçirdiği süreç ele alınacaktır.

Süreç içerisinde ele alınan kriterlerin belirlenmesi, kriterlere karşılık gelen verilerin elde edilmesi ve bilgiye dönüştürülmesi, elde edilen verinin tasarlanması, tasarımda kullanılan tekniklerin detaylandırılması ve bu sayede karar vericinin tahminleme yöntemi ile sahada aldığı olumlu-olumsuz her etkiyi önceden kavramlaştırmasına imkan verecektir. Kullanılan tekniklerin sisteme faydası performans analizi uygulamaları ile karşılaştırmalı olarak incelenecek ve yazılımın kalitesi arttırılmaya çalışılacaktır.

1.3 Hipotez

Karar vericinin aldığı kararların başarısızlık ile sonuçlanması, problemin doğru anlaşılmaması, analitik uygulamaların iyi yönetilememesi, tasarımın yetersiz kalması, verinin eksik tutulması gibi bir/birçok etkenin bir sonucudur. Karşılaştığımız her problemin çözümünde uygulanan yöntemler doğru karar verilmesine olanak sağlamakta ve bu sayede uygulanan tekniklerin gelişimi evrensel düzeyde önemli hale gelmektedir.

BÖLÜM 2

BANKACILIK SİSTEMİNDE KREDİ SÜRECİ VE KREDİ RİSKİ YÖNETİMİ

2.1 Kredinin Tanımı

Bankacılık sisteminde kredi kavramı, banka iç mevzuatları gereğince yapılan analiz çalışmalarına istinaden güven duyduğu kişi veya kurumlara nakit karşılığı belli bir süre ve bir bedel karşılığında satın alma gücünü kredi alana devretmesi olarak tanımlanabilir. Kredi oluşum süreci süre, güven ilişkisi, ödeyememe durumunda alınan risk ve karşılığında alınan gelirden ibarettir.

Belli bir süre sonunda geri alınması şartıyla ve krediyi geri ödeme konusunda kredi başvurusu yapan kişiye duyulan güven ile kredi verilmesine karşın her kredi kullandırımının geri ödenmeme riski vardır.

Teminatın yeterli olması, kredi verme kararının doğru tespit edilmesi ve kredi ihtiyacının sağlıklı analiz edilmesi kredi riskinin en aza indirilmesinde büyük önem taşır. Bir diğer açıdan finansal kuruluşlar için kredi müşterilerinden alınan komisyon ve faiz kalemleri gelir tablolarını etkileyen işlemler olarak bilinmektedir[23].

Yapısal özellikleri bakımından krediler ikiye ayrılırlar:

→Gayrinakdi tür krediler (Nakit olmayan krediler)

→Nakdi türdeki krediler (Plasman türündeki krediler)

Bu tez kapsamında ele alınacak kredi türü nakdi türdeki kredi türü olup taksitli bireysel kredi olarak sınıflandırılacaktır.

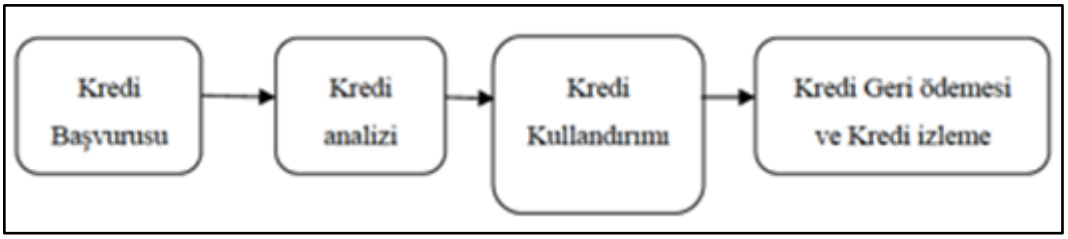
Faizi ile beraber belirli bir vadede geri ödenmek şartıyla gerçek veya tüzel kişilere bankalarca kullandırılan krediler nakdi türdeki krediler olarak tanımlanmaktadır. Nakdi krediler kurumsal ve bireysel krediler olmak üzere temelde ikiye ayrılır[24].

Taksitli bireysel kredi, ticari ya da sınai faaliyetlerin finanse edilmesi amacını taşımayan, kişisel ihtiyaçları gidermeye yönelik hizmet ve mal alımlarındaki finansman gereksinimi karşılamak üzere, gerçek kişilere taksitle geri ödenmesi koşuluyla kullandırılan kredi türüdür.

Bankalar müşteri ilişkileri de göz önünde bulundurularak kredilerin türüne göre çeşitli teminatlar talep edebilir, kredi verirken komisyon oranına bağlı ücret talep edebilirler.

2.2 Kredi Başvuru Süreci

Kredi verilme süreci müşterinin kredi talebi ile başlar ve bankaların iyi tanımlanmış başvuru değerlendirme kriterleri üzerinden kişilerin krediyi geri ödeme kapasitesini saptamasına yönelik analiz etme aşamasına tabii tutulur. Analiz aşamasında başvurunun kredi vermeye uygun bulunması durumunda ihtiyaca uygun tutar ve vadede kredilendilendirilmesine olanak verilir. Kredi kullandırımı ile beraber son ödeme yapılana kadar bloke konulan teminatların serbest kalması ile sonlanır.



Şekil 2. 1 Geleneksel Kredi Sürecinde Temel Aşamalar

Kredi başvuru sürecinde müşterinin krediyi geri ödeme durumunu önceden analiz etmesine imkan sağlayacak kriterlerin belirlenmesi ve bu kriterlerin kredi verme kararında uygulamaya alınması için uygulanan tüm iş analitiği teknikleri ve yazılım metodojleri bir mühendislik problemi olarak karşımıza çıkmaktadır.

2.3 Kredi Değerlendirme Analizinde Esas Alınan Kriterler

Kredi değerlendirme kriterleri belli bir müşteri portföy grubunda müşteri statüsüne-tipine(Emekli müşteri, Özel Sektör Çalışanı vb) bağlı olarak değerlendirilmektedir. Tüm müşteri statüsüne bağlı çalışan kriterler, kredi değerlendirme faktörleri olarak bilinmekte ve karar vermede etkinlik olasılığı yüksek değerler içermektedir.

Müşterinin net geliri, ödeme gücü, yaşı, medeni durumu, sahip olduğu menkulleri, kredi kartı kullanımı, ödeme sıkıntısına bağlı kanuni takibe düşme durumu, düzensiz taksit ödemeleri, firma geliri üzerinden kredi başvurusu yapması durumunda bağlı olduğu firmanın değerlendirme notu, KKB(Kredi Kart Bürosu) den alınan verileri(BBE, KKB puanı, kredi geçmişi, diğer bankalardaki kredi ödeme potansiyeli durumu gibi) kredi verme sürecine etki eden kriterlerin başvuru değerlendirme sürecinde analiz edilmesiyle başvuru değerlendirilmesi sonuçlandırılır.

Bu değerlendirme sürecinde karar verici ve başvuru talebinde bulunan müşteri, sistemin vereceği tepkiyi ve sistemden sağlayacağı faydayı anlamlandırmış olacaktır.

2.4 Kredi Değerlendirme Kriterinde Kullanılan Veri Seti Değişkenleri

Süreç içerisinde kriterlerin analiz edilmesinde gereksinimler dahilinde veri setinin oluşturulması karar verme sürecinde önemli etkiye sahiptir. Literatürde, “UCI Makine Öğrenim Deposu”nda bulunan veri kümesi üzerinden incelemeler yapılmaktadır.

Makine öğrenimi algoritmalarının analizi için makine öğrenme topluluğu tarafından kullanılan alan teorileri, veri tabanları ve veri üreteçleri topluluğu “UCI Makine Öğrenim Deposu” olarak bilinmektedir. Araştırmalarda açık kaynak olarak Alman kredi veri seti kullanılmış ve UCI Makine Öğrenim Deposunda da bulunan veri kümesi, bilgi keşfi sürecinden geçirilmiştir. Verinin kullanılabilmesi için veri dönüştürme ve ön işleme tekniklerini WEKA yazılım ara yüzleri kullanılarak anlamlandırmıştır.

Anlamlı hale gelen data üzerinden sınıflandırma algoritmaları (YSA,Lojistik regresyon vb) ve değerlendirme kriterleri(Gini Katsayısı, Duyarlılık, Kesinlik ve ROC değerleri vb) kullanılarak analiz edilen veri setinin doğruluğu tahmin edilmektedir.

Bu tez kapsamında kullanılan veri seti değişkenleri:

- **Başvuru Yapan Müşteri Bilgileri:** Kredi başvurusu yaptığı finansal kuruluşa ait müşteri numarası, kredi başvuru numarası ve müşterinin tüm özlük ve demografik bilgileri, müşterinin statü/tipi(kamu çalışanı, emekli müşteri tipi ..), müşterinin belgelendirilmiş maaş bilgisi, müşteri eş bilgileri, müşterinin yaşı, ikametgah adres bilgisi, iletişim bilgileri vb.
- **Model Oluşturulurken Kullanılacak Filtreler:** E –haciz/haciz kaydı, yasaklı çek durumu, donuk kredi durumu, takibe atılma durumu, yakın izlemeye alınma durumu, gecikme durumu, itfa durumu, hesap bloke durumu, varlık yönetim şirketine devir olmuş hesapları(VYŞ) gibi istihbarat bilgileri
- **Müşterinin Limit Risk Bilgileri:** Başvuru anı itibarı ile açık nakdi kredi sayısı, başvuru tarihi itibariyle nakdi riski, başvuru tarihi itibariyle nakdi limiti, başvuru tarihinden önceki son 6 ayda kullanılan (açık olan) nakdi ürün sayısı, başvuru tarihinden önceki son 6 ayda yapılan nakdi ürün başvurusu sayısı, başvuru tarihinden önceki son 6 ayda tahsis edilen nakdi ürün sayısı, başvuru tarihinden önceki son 6 aydaki ortalama nakdi riski, başvuru tarihinden önceki son 6 aydaki ödenen ortalama nakdi riski vb. limit/risk bilgileri
- **Müşterinin Gecikme Bilgileri:** Başvuru tarihinden önceki son 6 ayda var olan kredi ödemesinin geciktiği maksimum gecikme gün sayısı, başvuru tarihi itibariyle en son gecikmeye girilen gün üzerinden ne kadar gün geçtiği , başvuru tarihinden önceki son 6 ayda 30 gün ve üzeri gecikme sayısı , başvuru tarihinden önceki son 6 ayda 60 gün ve üzeri gecikmeye girmiş toplam tutar vb bilgileri
- **Müşterinin Hesaplarına İlişkin Bilgiler:** Başvuru tarihi itibariyle vadesiz mevduat hesaplarındaki toplam TL tutarı, başvuru tarihi itibariyle vadesiz mevduat hesaplarının sayısı, başvuru tarihinden önceki 6 ayda ortalama vadesiz mevduat tutarı TL, başvuru tarihinden önceki 6 ayın kaçında aktif hesap kullanımı yapıldı vb.
- **Müşterinin Teminat Bilgileri:** Başvuru tarihi itibariyle krediler için bağlanmış toplam teminat tutarı, başvuru tarihi itibariyle krediler için bağlanmış belli teminat tipleri için(teminatsız-açık, nakdi karşılığı, gayrimenkul karşılığı, taşıt karşılığı, kefalet karşılığı) teminat sayısı vb.

- **Müşterinin Çeklerine İlişkin Bilgiler:** Başvuru tarihi itibariyle son 1 ayda ibraz edilen çek sayısı, başvuru tarihi itibariyle son 1 ayda ibraz edilen çek tutarı, başvuru tarihi itibariyle son 1 ayda ibrazında ödenen çek sayısı, başvuru tarihi itibariyle son 1 ayda ibrazında ödenen çek tutarı vb.
- **KKB, Memzuç ve Diğer Finansal Bilgileri:** Başvuru tarihi itibariyle en güncel KRM(Kredi Risk Merkezi) sorgusunda müşterinin finansal sistemdeki toplam nakdi limit tutarı TL, başvuru tarihi itibariyle en güncel KRM sorgusunda müşterinin finansal sistemdeki toplam risk tutarı TL, başvuru tarihi itibariyle en güncel KRM sorgusunda müşterinin finansal sistemdeki toplam nakdi risk tutarı TL, başvuru tarihi itibariyle en güncel KRM sorgusunda ilk ibraz edilen çekin ibraz tarihinden başvuru tarihine kadar olan süre (ay), Dönen Varlıklar (başvuru tarihinden önceki aynı yıl içindeki en güncel mali bildirimden), diğer finansal kuruluşlardan alınan kredi kartı bilgisine istinaden limit boşluğunun belirlenmesi vb.

2.5 Kredi Risk Analizi

İlerleyen teknoloji çağında finansal kuruluşların yaşadıkları en büyük problem kredi riski ve analiz edilmesidir. Finansal kuruluşlar alacakları kararda kredi riskini en aza indirmek için kendi bünyesinde denetim ve gözetimlerini arttırarak önlem almaktadırlar. Finansal kuruluş olan bankaların da varolan risklerini iyi analiz etmesi ve oluşabilecek risklerini belirlemesi kredi süreçlerini olumlu yönde yönetebilmeleri için önemlidir. Günümüz çağında kredi risk yönetim sürecinde kullanılan temel metrikler ve kümülatif ilerleyen uygulamalar aşağıdaki gibidir[25]:

- **Kredi Risk Politikalarının Kontrol Edilmesi:** Finansal kuruluşlar kredi risk politikalarını ve stratejilerini belirli periyotlarla takip ederek aldıkları riske karşılık kazandıkları karı gözden geçirmelidirler.
- **Kredi Değerlendirme Analizinde Kullanılacak Kriterlerin Bulunması:** Finansal kuruluşların kredi değerlendirme kriterlerinin güvenilir olması hem analizin değerlendirilmesinde hem de kriterlerin kredi verme ölçütü olarak uygun çalışmasında önem arz etmektedir.

- **Kredilerin Risk Yönetiminin İzlenmesi:** Finansal kuruluşlar olası risklere karşılık izleyecekleri politikaları belirlemeli ve kredi sözleşmelerinde bu prosedürlere yer vermelidirler.
- **Kredi Risklerinin Raporlanması:** Finansal kuruluşlar kredi verirken ve verdikten sonra aldıkları riski diğer finansal kuruluşların kendi risklerini belirlerken kullanabilmesi adına sürekli gelişim içerisinde yer alan ilgili birimlere rapor etmelidirler.

Kredi riskini en aza indirmek ve riski tespit etmek adına uygulanması gereken ilkelerin başında kredi skorlama sisteminin varlığı gelmektedir.

2.6 Kredi Skorlama Tanımı ve Oluşturulması

Kredi skorlama başvuru sahibinin güvenilirliğini krediyi geri ödeyememe durumuna göre ölçen bir yöntemdir. Kredi riskinin değerlendirilmesinde en önemli ölçütlerden biri olan skorlama yöntemi, başvuru sahibinin ileriki dönemlerde krediyi geri ödeme durumunun nasıl bir ödeme performansı içerisinde gerçekleştireceğini tahmin etmeye yaramaktadır[26].

Skorlama mantığı; riski yüksek olan müşteriler için kredi verilmesini red ederek söz konusu finansal kuruluşun olası bir risk durumunda zarar görmesine engel olacak, riski düşük olan müşteriler üzerinden kredi vererek karını artırmasını sağlayacak ve müşterilerin de ödeme güçlüğü çekmeden kredi almasına imkan verecektir[27].

Kredi başvurusunda, müşteriden alınan bilgiler ile kredi değerlendirilmesinde kullanılan kriterlere karşılık gelen parametrelerin karşılığı alınır ve kriterlere karşılık gelen katsayılar üzerinden ağırlıklı çarpanlar alınarak skor puanı hesaplanır. Katsayılar ve çarpan ağırlıkları finansal kuruluşun daha önceden kullandırdıkları kredilerin analizi ile elde edilmektedir. Karar vericinin en büyük problemi olası bir değişiklikte kredi verme eğiliminin nasıl davrandığını tahminlemek ve alacağı kararların sahada önceden etkisini gözlemleyebilmek olmaktadır. Karar vericinin kredi vermede uygulayacağı handikapların ne olacağını belirlemek için öncelikle sahadaki etkisini görmesi gerekmektedir.

Bu amaçla kredi skorlama kriterlerine karşılık gelen değerlerin simülasyon çalışması ve raporlanması sonrasında da devreye alınması işin fonksiyonel kısımlarını oluşturmaktadır. İhtiyaç belirlendikten sonra fonksiyona hizmet edecek teknik mantalitenin kurulması ise ele alınacak çalışmanın mantıksal boyutudur.

Tüm bu etkenler göz önünde bulundurulduğunda karşımıza çıkan başlıklar: Analitik yaklaşım ile mühendislik konusunun irdelenmesi, problemi anlamak ve çözüm yöntemlerini araştırmak, uygulanacak iş analitiği ve yazılım metodojilerini oluşturmak ve yazılım geliştirmesini tamamlamak, uygulamanın sonucunu simüle etmek ve raporlamak olacaktır.

2.7 Kredi Skorlama Metotları

Geçmiş deneyimlerin analiz edilerek kredi verme eğilimini tahminleyebilmek için tahmin edici modellemeler aracılığıyla kredi skorlama yöntemleri kullanılmaktadır.

Yöntem olarak farklı metotlar ve algoritmalar kullanılmakla birlikte en yaygın olarak kabul gören istatistiksel metot regresyondur.

Gerek kredi skorlama yöntemleri olsun, gerekse skorkartların oluşum sürecinde olsun farklı metotların kullanılmasıyla birlikte günümüzde parametrik olan ve parametrik olmayan metot kavramı karşımıza çıkmaktadır. Parametrik olmayan metotlarda (K en yakın komşu yaklaşımı, yapay sinir ağları, karar ağaçları)kullanılan veriler üzerinde herhangi bir varsayım yapılmazken, parametrik metot(lineer regresyon, diskriminant analizi, lojistik regresyon)kullanılan veri üzerinde bazı varsayımlar kabul edilmektedir [28].

Kredi skorlamada en sık kullanılan geleneksel istatistik tekniklerin, diskriminant analizi ile lojistik regresyon analizi olduğu söylenebilir[29].

Genellikle k-en yakın komşuluk metodu, karar ağaçları ve doğrusal programlamanın da kullanılabildiği kredi skorlama problemlerinde, son yıllarda YSA'lar, uzman sistemler ve yapay zeka teknikleri de kullanılmaya başlanmıştır. Ayrıca bu tekniklerin kombinasyonlarının uygulandığı hibrit yaklaşımlar da mevcuttur.

2.7.1 Diskriminant Analizi

Diskriminant analizi ayırma ve sınıflama aracı olarak ilk kez Fisher tarafından önerilmiştir[30].

Gözlemlerin gruplar arası varyansını grup içi varyansına oranını maksimum yapmaya çalışan bir istatistik tekniğidir. Fisher ilk olarak çiçeklerin belli özelliklerini kullanarak hangi gruba ait olduğunu bulmaya çalışmıştır[31].

Çok değişkenli istatistik tekniklerinden biri olan diskriminant analizi ile çekildiği kitle bilinmeyen birimlerin minimum hata ile sınıflandırılması amaçlanır. Bu analizin temelinde birimlerin sınıflandırılmasını sağlayan eşitlik (2.1)'de gösterildiği gibi bir fonksiyon yatar[32]:

$$Y_i=a_ix_{i1}+a_2x_{i2}+...+a_px_{ip} \quad (2.1)$$

Diskriminant (ayırma) fonksiyonu olarak adlandırılan bu yapı sayesinde birimler iki ya da daha fazla sayıda gruba yerleştirilebilir. Bu fonksiyon bulunurken oluşturulacak olan grupların ortalamalarının maksimum derecede farklı olması istenir. Diskriminant analizinin uygulanabilmesi için; bağımsız değişkenlerin birbiri arasında çoklu doğrusal ilişki ya da doğrusala yakın ilişki probleminin olmaması, grupların ortak kovaryans - varyans matrisine sahip ve verilerin çok değişkenli normal dağılıma sahip olması gerekmektedir.

Diskriminant analizi, bireylerin en az hata ile ait olduğu kitleyi ayırmak için kullanılan kümülatif işlemler olarak belirtilebilir. Lineer diskriminant analizinin zayıf tarafı genellikle lineer olmayan değişkenler arasındaki lineer ilişki varsayımının varlığı ve çok değişkenli normal dağılım varsayımından gelen sapmalara karşı duyarlı olmasıdır[29].

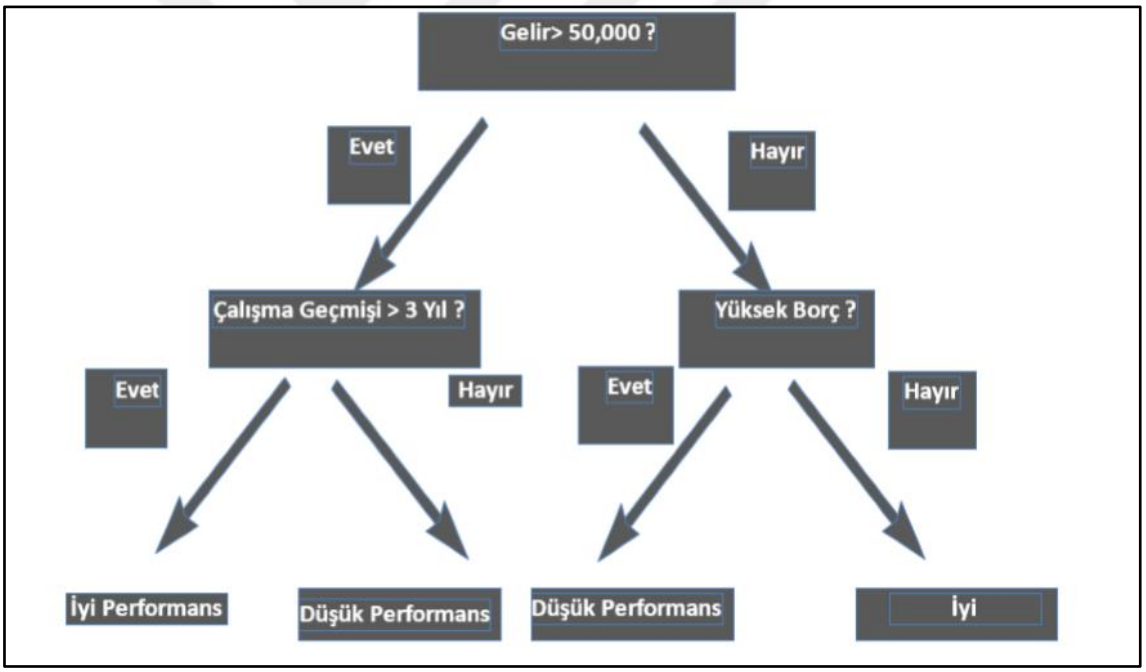
2.7.2 Lojistik Regresyon Analizi

Bu teknik, aralarında herhangi bir bakımdan benzerlik bulunan bağımlı değişken gruplarının olduğu durumlarda kullanılabilir. Lojistik regresyon analizi çeşitli varsayımların(ortak kovaryansa sahip olma, normallik gibi) açıkça bozulumu durumunda çapraz tablolara ve diskriminant analizine bir seçenek olmaktadır.

Ayrıca, bağımlı değişkenin ikili (0 ve 1 gibi)ya da ikiden fazla düzey içeren kesikli değişken olması durumu yani normallik varsayım kısıtı olmaması, çözümlenmeyen modelin matematiksel açıdan kolay yorumlanabilir ve çok esnek olması olması yöntemin tercih edilme sebebidir. Bağımlı değişkenin ikiden çok düzeye sahip kesikli değişken olması bu değişkenin açıklayıcı değişkenler üzerindeki regresyon modeli, çoklu grup (multigroup-polychotomous) lojistik model adını almaktadır[32].

2.7.3 Karar Ağaçları

Her küme için bir aktivite seçiminin söz konusu olduğu, birden fazla olaylar kümesinin bulunduğu çok kademeli bir süreçtir. Karar ağacı, ilk olaydan belirli bir stratejiyle son sonuçlara ulaşılma aşamasına kadar ortaya çıkan tüm olay ve eylemlerin tarih sırasına göre düzenlenmesiyle oluşan bir grafiktir. Grafik ağaca benzediğinden "ağaç grafiği" veya "karar ağacı" denmektedir[33].



Şekil 2. 2 Karar ağacında kök ve düğüm yapısı [19]

2.7.4 Doğrusal Programlama

Doğrusal bir amaç fonksiyonu, iyi tanımlanmış doğrusal eşitsizliklerin veya eşitliklerin sınırlandırılmış koşulları altında, en optimum kılan değişken değerlerinin belirlenmesinde kullanılan matematiksel teknik olarak bilinmektedir [34].

Doğrusal programlama geniş bir uygulama alanına sahip olmakla birlikte, bir sınıflama yaklaşımı olarak da kullanılabilir. Varsayalım ki, bir kredi başvuru formunun cevaplarından yola çıkarak oluşturulan bir örnekte n_G tane iyi, n_B tane kötü kredi başvuru sonucu, m tane karar değişkeni ve i tane başvuran kişi (X_{i1} , X_{i2} , ..., X_{im}) olsun.

Bir sınır değeri c ile belirtilsin ve bu değerin üstünde bir değere sahip olanların "iyi", altında bir değere sahip olanların ise "kötü" olarak nitelendirildiği bir lineer skorkart geliştirilmeye çalışılsın. Burada "a_i" olası hata değerlerini ifade etmek üzere, bu hataların toplamını minimize etmek için kullanılan ağırlıkların (W_1 , W_2 , ..., W_m) en iyi değerlerinin bulunmasının amaçlandığı bir doğrusal programlama problemi aşağıdaki gibi modellenebilir [35]:

$$\min \; a_1 + a_2 + \ldots + a_{nG+nB}$$

$$W_1X_{i1} + W_2X_{i2} + W_mX_{im} \geq c - a_i \quad , \quad 1 \leq i \leq n \qquad (2.2)$$

$$W_1X_{i1} + W_2X_{i2} + W_mX_{im} \leq c + a_i \quad , \quad n_G + 1 \leq i \leq n_G + n_B \; , \; a_i \geq 0 \; , \; 1 \leq i \leq n_G + n_B$$

Hem lineer hem de lineer olmayan yapıları sınıflamada doğrusal programlamanın kullanılabileceği ilk kez Mangasarian tarafından gösterilmiştir[36]. Daha sonraları benzer şekilde tamsayılı programlama yöntemiyle de kredi skorlama problemleri ele alınmıştır.

2.7.5 Karar Destek Sistemleri

'KDS – Karar destek sistemleri' bir kurumda profesyonel çalışanların ve yöneticilerin karar vermesini sağlamak için kullanılan, bu süreçte kullanıcı ve sistemin karşılıklı olarak etkileşim halinde bulunduğu bilgisayar tabanlı bilişim sistemi olarak tanımlanabilir. KDS dört karakteristik özelliğe sahiptir.

Sistem yapılandırılmamış veya yarı-yapılandırılmış karar verme amacına hizmet etmek amacıyla tasarlanmıştır ve otomatize edilmiş kararlar dışındaki karar verme durumlarında da yardımcı olmaktadır.

Karar destek sistemleri, karar veren kişinin değişen ihtiyaçlarını da kısa sürede cevaplayabilir, karar vermenin verimliliğini artırmakla birlikte etkinliğini de artırır [37].

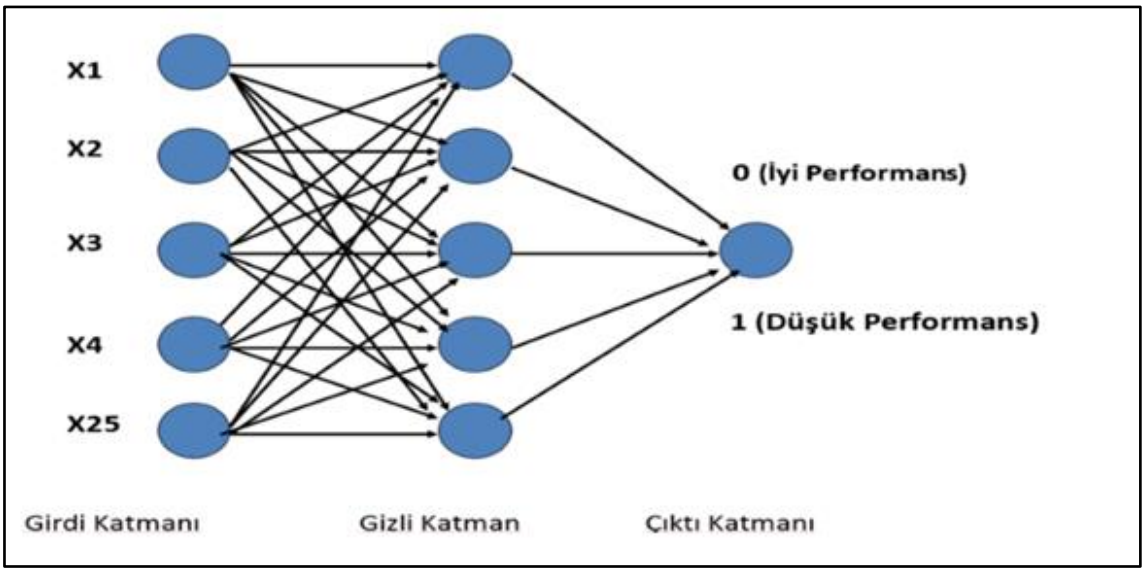
2.7.6 Yapay Sinir Ağları

Değişkenler arası ilişkilerin lineer olmadığı varsayımı altında, bir grup girdi ile çıktı arasındaki ilişkiyi modelleyebilen YSA'lar, son zamanlarda kredi skorlama için etkili bir yöntem olarak kullanılmaktadır[38].

YSA'ların önemli özelliklerinden birisi, verilerle ilgili bir takım kabulleri gerektirmiyor olmasıdır. Örneğin, istatistiksel esaslı yöntemlerin hepsinde verilerin veya model sonrası hataların normal (Gauss) dağılımına uyması zorunluluğunun yanında parametrelerin zamanla sabit olması, paralel işlemler yapılamaması, bağımlılığın doğrusal olması, vb. kabuller mevcuttur.

Halbuki geleneksel YSA'larda bu tür kabuller yoktur ve uygulanmaları için paralel işlemlere meydan verebilecek en azından biri girdi, diğeri çıktı ve gerekirse de bir diğeri de ara (saklı) olacak biçimde üç tabakanın hücrelerle beraber tesis edilmesi gereklidir. Bunun için sadece matematik kurallar değil; etki - tepki değişkenleri ile kendi içinde oluşabilcek tepkimeleri modelleyebilecek mimarinin geliştirilmesi gerekmektedir [39].

Son zamanlarda kredi skorlama modelleri oluşturulurken yapay zeka (YZ) teknolojilerinden sıklıkla yararlanıldığı görülmektedir. Literatürde, kredi skorlama problemlerinin çoğunda YSA'ların diskriminant analizi ile lojistik regresyon analizinden daha iyi bir performansa sahip olduğu görülmüştür[40] .



Şekil 2. 3 Kredi Skorlamada YSA Yapısı [19]

BÖLÜM 3

KREDİ SKORLAMA SİSTEMİNDE KULLANILAN İŞ ANALİTİĞİ TEKNİKLERİ

Kredi skorlama sisteminin nasıl çalıştığı çalıştığının ve davrandığının yol haritası mühendislik yaklaşımı olan iş analizi teknikleri ile anlatılacaktır. Uygulamada kullanılan birçok modelleme olmakla beraber bu çalışma kapsamında sistemin davranışını sergileyen temel modellemeler üzerinden sınırlandırma yapılacaktır.

İş analitiği, veriyi iç görüye dönüştürerek daha iyi karar vermeyi sağlayan bilimsel sürecidir. Sürecin bir aşamasındaki başarısızlık, sonraki adımların da başarısız sonuçlanmasını doğurur. Örneğin, analiz aşamasındaki eksik tanımlama yazılım-test aşamasını olumsuz etkiler. Bu nedenle iş analitiği tekniklerinin yer aldığı projeler için; işin tanımlanması(kapsamın detaylı hazırlanması), analizlerin yapılması, programın tasarım sürecinin iyi kurgulanması, test edilmesi ve tüm dokümantasyonların oluşturulması aşamalarını kapsamaktadır.

3.1 Analitik Yaklaşım ve Metodolojileri

Analitik yaklaşım, problem çözmeye dayalı ve keşif amaçlı olmak üzere iki farklı paradigma üzerinden veriyi elde ederek ve gruplara ayırarak sistemi irdeler.

Problem Çözmeye Dayalı:

- Tanımlanabilir bir problemin varlığı
- İş problemi ile ilgili hipotezlerin olması
- Veri kaynaklarının kullanılarak probleme çözüm oluşturulması
- Karar ve aksiyona dönüşen içgörü ve önerilerin yaratılması

Keşif Amaçlı :

- Amaç verinin keşif amaçlı incelenmesi
- Veride gizli ilişki ve örüntülerin keşfi ile içgörü yaratmanın hedeflenmesi
- Nihai amaç; iş fırsatları ve iyileştirme imkanı sağlayacak problemin keşfi

Problem tanımında farklı yaklaşımlar üç grup altında toplanabilir:

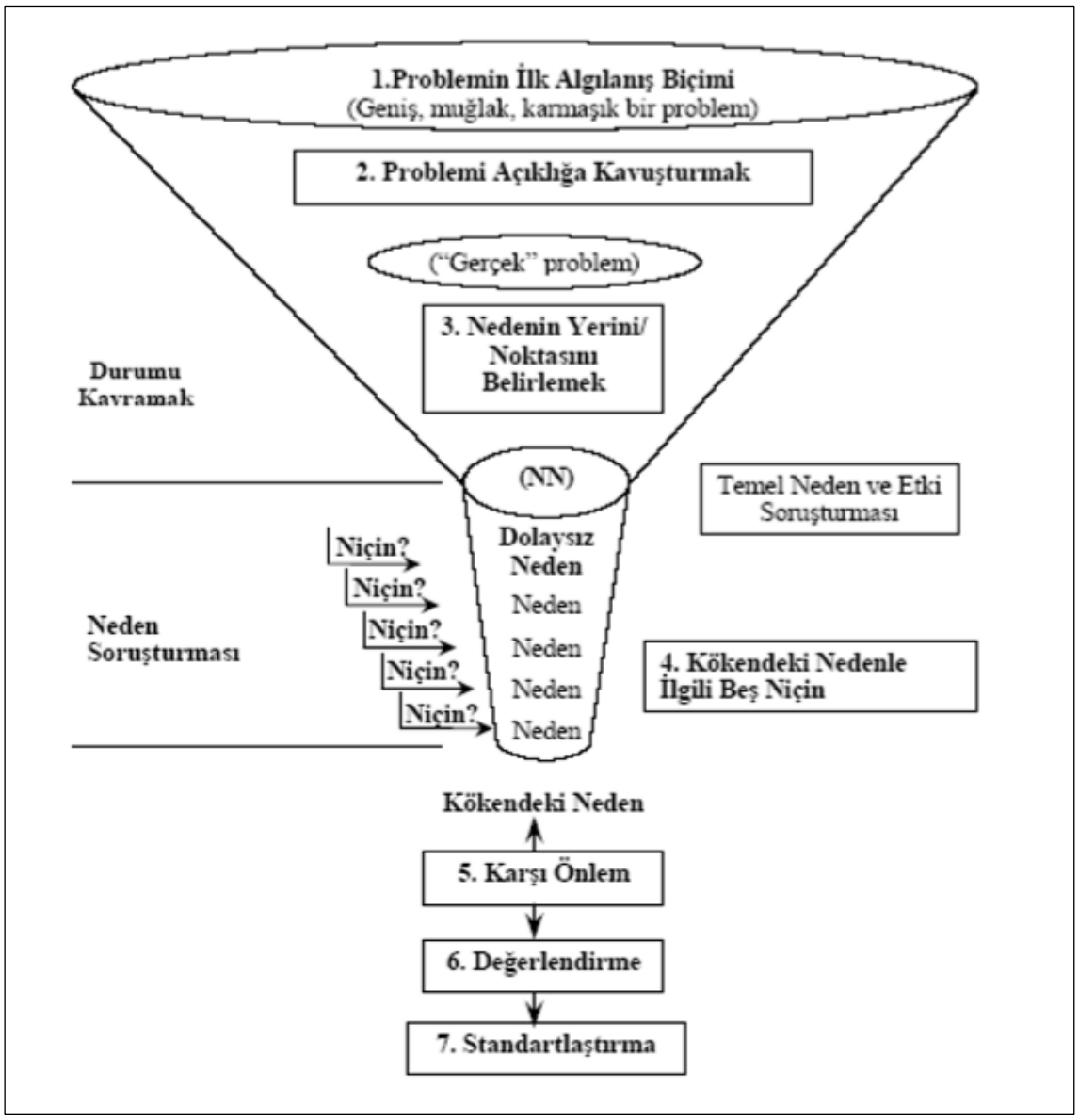
- Kompleks problemleri tanımlayan genel ifadeler kullanılabilir. “Kredi kullandırımında bankanın batık kredi vermemesi için alacağı önlemler ne olmalıdır?” örnek verilebilir.
- Genel ifadelerle problemi ifade ettikten sonra daha spesifik bir ifadeyle yeniden formüle edilebilir. “Banka kredi kartı verme kriterleri üzerinden hangisinde değişiklik yaparsa kredi kartı satışında olumlu yönde kar elde etmesi mümkündür” şeklinde genel olarak ifade edilir; sonrasında daha dar bir anlamda ifade edilir. “Maaş müşteri kitlesinde değişiklik yapılmalıdır”. Daha sonra emekli müşteri tipinde değişiklik yapılmasına karar verilir ve nihayetinde “Bankamız emekli maaş müşterileri özelinde ....” değişiklik yapılması olarak ele alınır.
- İşe bir problem üzerinden başlanır ve problem çok dar kapsamlı hale getirilir. Sonrasında problem adım adım genişletilerek çıkarılan her ayrıntı bu genişlendirme sürecinde ele alınır.

Problemi tanımlarken konu, kapsam ve amacın tanımlanması sürecin olumlu çıktı üretmesi açısından (problemin tespiti) gereklidir:

Konu; sistem analizinin veya problemi tanımlamanın merkezinde yer alır.

Kapsam; problem ya da konu tanımlandıktan sonra, çalışmanın içeriği ve sınırlarının belirlenmesi anlamına gelmektedir.

Amaç; asıl gerçekleşmesi istenen başarıdır. Hem konu hem de kapsam dahilinde olmalıdır.



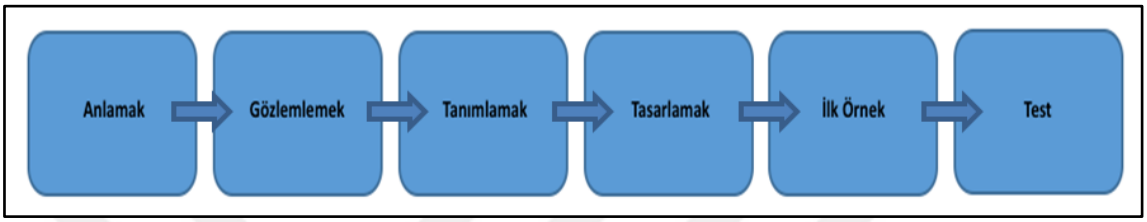
Şekil 3. 1 Pratik Problem Çözme Süreci [41]

3.1.1 Tasarım Odaklı Düşünme

Herbert Simon, 1969 yılında "Yapay Bilimler" adlı çalışmasında "tasarım düşünce" sürecinin resmi olarak ilk modellerinden birini ele almıştır. Model her biri aktivitelere ve bileşene sahip yedi ana kademeden oluşmakta ve günümüz "Tasarım Düşünce" sürecinin temelini oluşturmada öncülük etmektedir. Tasarım odaklı düşünme çözüm esasına dayalı tasarım metodolojisi yaklaşımıdır.

Tasarım odaklı düşünme ile bilinmeyen kompleks problemlerin ya da kötü tanımlanmış problemin üstesinden gelmek, var olan ihtiyaçları belirleyerek sorunu yeniden ölçeklendirmek, beyin fırtınası oturumlarında fikirler üretmek ve uygulamalı bir yaklaşım benimsemek son derece faydalı olmaktadır.

Tasarım odaklı düşüncenin aşamalarının bir problem üzerinden kullanılması, etrafımızda meydana gelen kompleks problemlerin çözülmesi için bizleri, "Tasarım Odaklı Düşünce" metotlarının kullanımına yönlendirecektir[41].



Şekil 3. 2 Tasarım Odaklı Düşünce

Anlamak: Tasarım süreci için anlamak, çözmeye çalıştığınız problemin tarafsız ve değerlendirmeli bir anlayışını kazanmaktır. Anlamak bazı kaynaklarda empati kurmak olarak da karşımıza çıkar, kullanıcılar ve onların ihtiyaçlarına yönelik içgörü kazanmaları için kendi dünyayla ilgili kendi varsayımlarını bir kenara bırakmalarını sağlar.

Gözlemlemek: Anlayarak ortaya konulan bilgilerin bir araya getirilerek temel sorunları tanımlamak için gözlemlerin analiz edilip bunların sentezlenmesi gerekmektedir.

Tanımlamak: Gözlemleme sonucunda problemin tanımlanması; tasarımcıların sorunları, fonksiyonları ve problemleri çözmelerine ya da en azından zorlukları en az sorunla çözmelerine olanak tanıyan diğer unsurları oluşturmak için fikir toplamasına yardımcı olmaktadır.

Tasarlamak: Problemin anlaşılması, gözlemlenmesi ve tanımlanması sonucunda tasarım oluşturulma aşamasına gelinir ve hiç kuşkusuz temelleri iyi oluşturulan tasarımın sahada ilk prototipi başarı ile sonuçlanır.

İlk Örnek: Tasarım sonucu ürün içinde bulunan belirli özellikleri üretecek, böylece önceki aşamada üretilen problem çözümlerini araştırabilecek ilk Prototipler oluşturulur. Amaç, ilk dört aşamada tespit edilen sorunların her biri için mümkün olan en iyi çözümü belirlemektir.

Test: Tasarımcılar veya değerlendiriciler, prototipleme aşamasında belirlenen en iyi çözümleri kullanarak ürünü eksiksiz bir şekilde test eder.

Bu 6 aşamalı modelin son aşamasıdır, ancak yinelemeli bir süreçte, test aşamasında elde edilen sonuçlar genellikle bir veya daha fazla problemi yeniden tanımlamak ve kullanıcıların anlayışını, kullanım koşullarını, insanların nasıl düşündüklerini bildirmek için kullanılır.

3.1.2 Problem Belirleme Yöntemlerini Kullanma

İş analitiği tekniklerinin uygulamadan önce problemleri bileşenlerine ayırmak için farklı problem çözme metodolojilerinden ve bunların araçlarından yararlanılabilir. Bunlardan bazıları: Deneme-Yanılma Yöntemi, Neden Sonuç Analizi, Balık Kılçığı Yöntemi, Altı Sigma Yöntemi, SWOT Analizi, Karşılaştırma Analizleri, Performans Raporları, Alan Araştırmaları, Veri Analizi .. gibi değişik yaklaşımlar kullanılabilir.

3.1.2.1 Deneme-Yanılma Yöntemi

Bir problemin hemen çözülmeye çalışılması ve problemin tanımlanmasının hemen akabinde çözümünün ortaya koyulması yaygın bir davranıştır. Eğer üretilen fikir işe yaramıyorsa, işe yarar bir çözüm elde edene kadar yeni bir fikir üretilmelidir. Bu problem çözme tekniği, Deneme-Yanılma yöntemi olarak adlandırılmaktadır[42].

Bu tez kapsamında ele alınan ‘skorlama sisteminin kredi alma sürecindeki etkisi’ konusu altında sistematik yapı kurulmadan önce deneme-yanılma yöntemi üzerinden bilgi edinilmiştir.

Deneme-yanılma yönteminin özelliklerini aşağıdaki gibi sıralayabiliriz[43]:

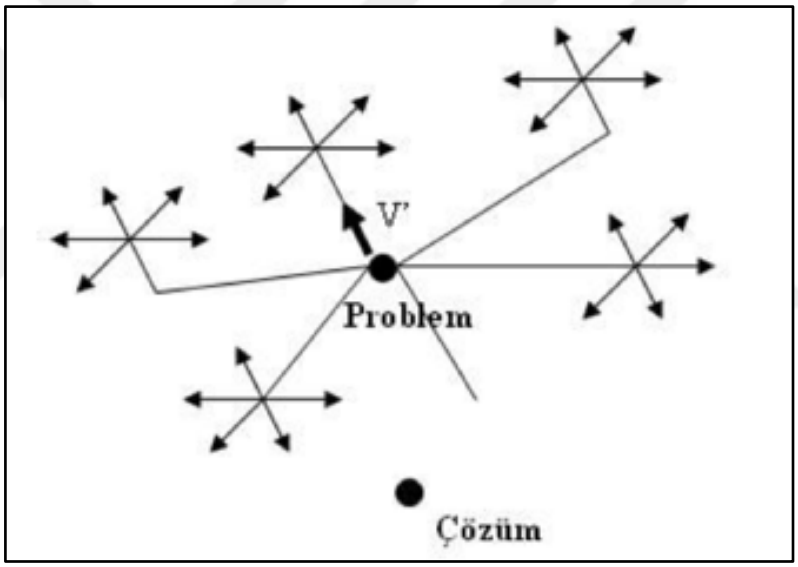
*Çözüme yöneliktir: Bir çözümün neden işe yaradığını keşfetmeyi denemez, sadece çözümle ilgilenir.

* Probleme özgüdür: Bulunan çözümü diğer problemlere genelleştirmeyi denemez.

* Optimal değildir: Deneme-yanılma bütün çözümleri ve en iyi çözümü değil; sadece bir çözüm bulmaya yönelik bir denemedir.

* Az bilgi birikimi yeterlidir: Konu hakkında az bilginin bulunduğu veya hiçbir bilginin bulunmadığı durumlarda devam eder.

Yaratıcı kişi, problem noktasından son konumu belli olmayan sonuç noktasına ulaşmayı hedefler. Bunun için bir doğrultuda araştırma kavramı geliştirir ve deneme yaparak ilerler. Yanlış kavramın seçimi ve başarısızlık, yeni bir kavramın ortaya konulması ve başlangıç noktasına dönülmesi sonucunu ortaya koyar. Sonuç bulunana kadar devam eden bu sürecin deneme sayısı oldukça fazla olmaktadır. Bu yöntemin belirgin bir özelliği aşağıdaki şekilde gösterildiği gibi birden fazla gidiş okunun çözümün tersi yönünü göstermesidir. Başlangıçta, yaratıcı kişinin deneyimlerine göre Atalet Vektörü (V') yönünde ilerleyen ve karmaşık olmayan denemeler, süreç boyunca sayıca artarken, kişinin problemden uzaklaşmasına da sebebiyet vermektedir [44].



Şekil 3. 3 Deneme-Yanılma Yöntemi [53]

İlk bakışta karmaşık görünen deneme-yanılma yönteminin kendine ait bir sistemi vardır. Denemelerin sayısı arttıkça alışılmış bir yön bulmak kolaylaşır ancak bu yöntemin zayıf noktası yaratıcı kişinin bilinçaltının da etkisiyle bu yön boyunca ilerleyerek yeni keşif yapmasının zorlaşmasıdır. Kişi daha kolay çözümler üretebilecek noktaya gelmek yerine sadece karşısına çıkan engelleri bertaraf etmekle uğraşır ve kimi zaman bu yol içinde çözüme ulaşamadan kaybolur.

Deneme-yanılma yönteminde, problemin karmaşıklığına göre denemelerin sayısı değişecektir. Eğer çözüm, araştırmayı yapan kişinin uzmanlık alanındaysa denemelerin sayısı daha az olacaktır.

Aksi halde yaratıcı kişi kendi deneyim ve bilgisinin dışında yeni alanlara bakacak ve bu durumda denemelerin sayısı kişinin deneme-yanılma gibi psikolojik bir aracı ne kadar iyi kullandığına bağlı olarak değişecektir.

Bu, aranan çözümün yaratıcı kişinin kendi deneyimi doğrultusunda aranmasına ve yeni kavramlar geliştirmek üzere alternatif teknolojilere bakılmamasına, yani Psikolojik Atalete yol açmaktadır [45].

Sonuç olarak deneme-yanılma yöntemi hakkında aşağıdaki sonuçlara ulaşılabilir [44]:

1. Yaratıcı kişinin düşünceleri bilinen bir kavramdan bilinmeyen bir kavrama doğru yol alır. Önceden var olan bir araç, yaratıcı kişinin değiştirmeye çalıştığı ilk örnek olarak kullanılır. Bu durum bir dizi başarısız sonuç üretir. Atalet vektörü çözümden uzağa yönlendirme yapar.
2. Yaratıcı kişi doğru çözüm yönünde tamamen yeni bir yol seçmeye zorlanır. Bu yol da başlangıçta kendisi için bilinmezdir. Mantıksal ve güvenli bir şekilde bir başarısız fikirden diğerine bu yolda nasıl ilerlediğini açıklayabilir. Sonra yol aniden bitiverir. Mantıksal açıklamaların yerini fikirlerin kendisini nereye savurduğunu belirten açıklamalar alır.
3. Her ne kadar yaratıcı kişinin ulaştığı sonuç başarılı da olsa araştırması mükemmel olmaktan çok uzaktır.

3.1.2.2 Beyin Fırtınası

Amerikalı psikolog A. Osborn, 1953 yılında deneme-yanılma yöntemi üzerinde geliştirmeler yaparak fikir üretme ve analiz etme süreçlerini birbirinden ayırarak çözme yoluna gitmiştir. Bu iki süreç deneme yanılma yönteminde yaratıcı tarafından yürütülen farklı yetenekleri barındırmaktadır.

Beyin fırtınası açık bir soru ile başlar ve olgunlaşmamış iyi ve kötü özellikteki fikirlerin listesinin oluşturulmasıyla biter.

Beyin fırtınası sürecinde bu fikirleri analiz etmeye kalkmak sürecin bozulmasına yol açar. Süreç bittikten sonra diğer kalite geliştirme araçları ile fikirler analiz edilebilir.

Beyin fırtınası yapılandırılmış ve yapılandırılmamış olmak üzere iki şekilde uygulanabilir [46].

Yapılandırılmış Beyin Fırtınası: Bu yöntemde herkes rotasyonda sıra kendilerine geldiğinde bir düşünce üretmeli, ya da diğer tura kadar pas geçmelidir. Bu yöntem insanların katılımcılığını zorlayabilir, ancak katılımcılığın sağlanması yönünde bir miktar basınç da oluşabilir.

Yapılandırılmamış Beyin Fırtınası: Bu yöntemde grup üyeleri basit bir şekilde konu ile ilgili olarak akıllarına gelenleri söyler. Sıra baskısı yoktur, rahat bir atmosfer vardır, ancak çok konuşan üyelerin diğerleri üzerinde üstünlük kurmaları riskini içerir.

Beyin fırtınası sorunu ve sorunun çözümünü bulmaya yarayan bir yöntemdir. Gizli kalmış görüşleri açığa çıkarmayı ve farklı düşünmeyi amaçlamaktadır. Beyin fırtınasına katılan kişiler sorunları önem sırasına göre dizerler. Başarıya ulaşmak için sorunun anlaşılır olması, sorunun çözümünde tekrardan kaçınılması, sorun incelenirken detaya inilmemesi gerekir [47].

Beyin fırtınası sürecinde takımın bir üyesi bir fikir ortaya atınca, başka bir üye hemen bu fikir üzerinde değişiklikler yaparak yeni bir fikir üretmeye veya fikri geliştirmeye başlar. Bu noktada ilk üye de kendi fikrine başka bir bakış açısı ile bakabilmektedir. Her üyenin katılımıyla fikir gelişimi bir zincir haline gelecek şekilde devam eder. Ancak fikirlerin başarılı bir şekilde gelişmesini sağlayan bu mekanizma bazen çözümün zıt yönünde ilerleyebilir.

Başarılı bir beyin fırtınasını yönetmek için,

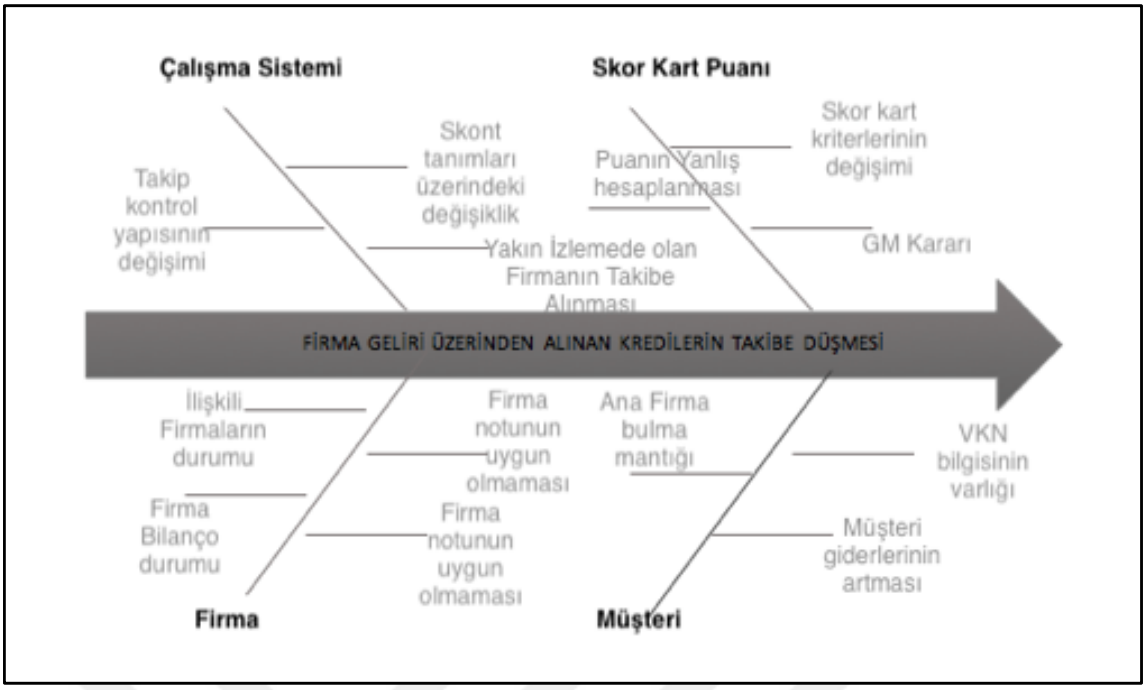
1. Fikirlere açılmadan önce katılımcı herkesin merkezdeki soruyu anlaması ve bu sorudan tatmin olması gerekir.
2. Başlamadan önce herkese birkaç saniye zaman vererek bazı fikirleri kağıda karalamalarına izin verilebilir.
3. Masanın ya da odanın etrafında dolaşarak ve herkese fikrini söyleme ya da pas geçme olanağı tanımak gerekir.

4. Fikirlerin çok olmasını engellememek, radikal fikirleri desteklemek gerekir.
5. Fikirler hakkında yargılamayı engellemek gerekir.
6. Söylenenleri tam olarak kaydetmek ve herkesin fikirleri bittikten sonra açıklığa kavuşturmak gerekir.
7. Fikir üretimi seyrekleşene kadar durmamak ve geç fikirleri reddetmemek gerekir.
8. Tekrarlamaları ve konu dışı fikirleri engellemek gerekir.

3.1.2.3 Neden-Sonuç Analizi (Balık Kılçığı Diyagramı)

İlk olarak Tokyo Üniversitesinde hocalık yapan Ishi Kawa tarafından, mühendislik okuyan öğrencilerine farklı faktörlerin birbirleriyle olan ilişkilerinin analizi için bir metod olarak sunulmuştur. Neden-sonuç analizi, operasyonel problemleri tanımlamada ve teşhisinde kullanılan bir araçtır. Bir sürecin çıktı yada sonucu birçok faktöre bağlıdır ve bu faktörlere yönelik olarak bir neden ve sonuç ilişkisi oluşturulabilir. Sürecin sistematik olarak gözlenmesi ile birçok neden ve etki ilişkisi belirlenebilir. Neden ve sonuç ilişkisinden oluşan bu yapıyı göz önüne almadan karmaşık sorunların çözümü zordur. Kalın dallar nedenleri ifade eder ve her temel nedenden üç ile altı arası alt neden üretilir [48].

Balık kılçığı diyagramı, ilk kullanıldığı zaman diliminde, Japonya genelinde büyük bir kullanım alanına ve hızlı bir şekilde yayılmıştır. Amacı ise, sonuçları meydana getiren çeşitli nedenleri göz önüne getirmek, bu nedenleri ve alt açıklamalarını görselleştirebilmek ve tüm bu nedenler üzerinde çalışarak sorunları en alt seviyeye indirmek için hazırlanır. Balık kılçığı adıyla bilinen sebep sonuç diyagramının genel uygulama alanı ise, ürün tasarımı ve kalite hatalarının engellenmesi hususları üzerindedir. Karşılaşılan problemi oluşturan bütün sebepler, bir takım birbirinden farklı sıkıntı kaynakları nedeniyle ortaya çıkar. Soruna götüren sebepler ise, genel olarak bu kaynakları tespit edebilmek maksadıyla, ana kategorilere ayrılarak yazılır [49].



Şekil 3. 4 Balık Kılçığı Diyagramı

Neden-sonuç analizi süreci altı aşamada özetlenebilir:

1. Problem olabildiğince doğru ve eksiksiz bir şekilde tanımlanır.

2. Tanımlanacak muhtemel neden için temel gruplamalar belirlenir. (yöntem, makina ve insan gücü en yaygın kullanılanlardır.)

3. Tüm üyeler beyin fırtınası oturumuna katılır. Sayfanın ortasına soldan sağa, sağ tarafta kutu içinde sonuç (sorun) yer alacak şekilde ana kılçık çizilir. Sonuca etki eden farklı kategoriler bu ana kılçık üzerine, yine kutu içinde olacak şekilde çizilir. Belirlenen diğer nedenler küçük kılçıklar olarak temel kılçıklara eklenir.

4. Tüm nedenler oylanır. Üyeler birden fazla nedene oy verebilirler.

5. En fazla oy alan nedenler işaretlenir. Üyeler muhtemel nedenler üzerinde tartışma açabilirler. Her türlü tartışma bittikten sonra ikinci oylamaya geçilir. Bu oylamada her üyenin tek oy hakkı vardır.

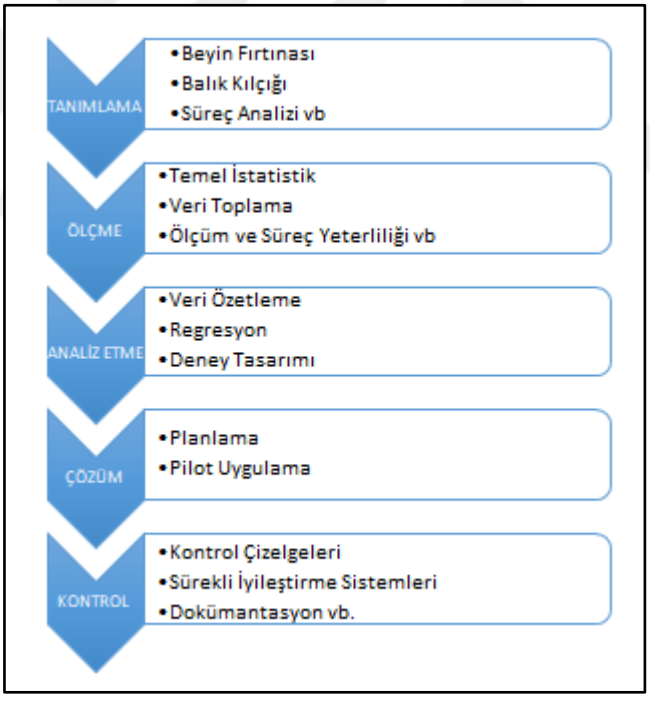
6. En fazla oyu alan neden, o soruna yol açıp açmadığını anlamak amacıyla doğrulanır.

3.1.2.4 Yalın Altı Sigma Yöntemi

Yalın Altı Sigma, müşteri tatmini, maliyet, kalite, süreç hızı ve yatırılan anaparanın iyileştirilmesinde en yüksek hızı yakalayarak; paydaşların elde edeceği değeri maksimize eden bir metodolojidir [50].

Herhangi bir süreçte hataları azaltmak ve yeterliliği artırmak için gerekli teknikleri sağlamak amacıyla kullanılır. Günümüzde bu teknik üretim, tıp, sigorta gibi birçok hizmet sektöründe kullanılmaktadır. Bu yöntemin temel prensibi kalitenin artmasının hataların azaltılmasının ön koşul olduğu kabulü vardır.

Altı Sigma teknolojisi herhangi bir iş sürecini, sürecin sürekli olarak gözden geçirilmesi ve yeniden ayarlanması ile geliştirmek için Olanakları Tanımlama, Performansı Ölçme, Olanakları Analiz Etme, Performansı Geliştirme, Performansı Kontrol Etme adımlarından oluşan bir yöntem kullanır.



Şekil 3. 5 Altı Sigma Süreci [51]

Yalın Altı Sigma çalışmalarının başarıya ulaşması için öncelikli kural ekibin, şirket ve müşterileri için önem taşıyan bir problem üzerinde çalışmasıdır [51].

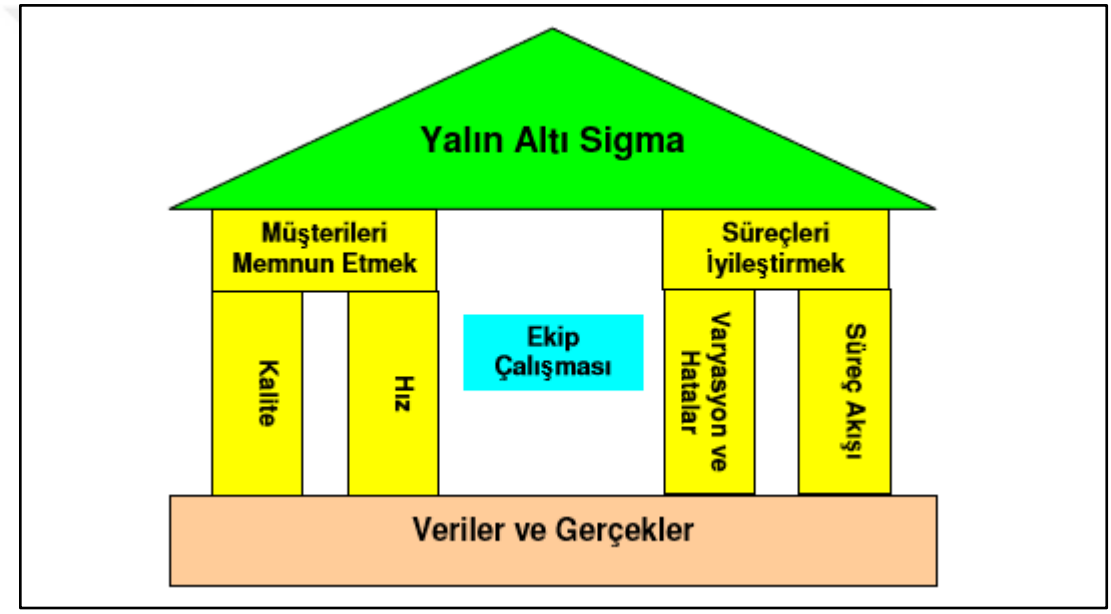
Ayrıca;

• Amaç müşteriyi memnun etmek olmalıdır, yani daha kısa sürede daha kaliteli hizmet vermektir.

• Bu hedefe ulaşmak için süreçleri iyileştirmek gereklidir. Bunu yapmak için varyasyon ve hataları (müşterinin kabul etmeyeceği her şey) ortadan kaldırılmalı ve işin süreçler boyunca nasıl ilerlediğine odaklanmaları gerekmektedir.

• Farklı süreç bölgelerinde çalışan kişiler ekip çalışması yapmalı ve problemi çözebilmek için fikirlerini birbirleriyle paylaşmalıdır.

• Verilen tüm kararlar verilere dayanmalıdır.



Şekil 3. 6 Yalın Altı Sigma'nın Anahtarları [51]

Çözümleri oluşturmak için birlikte çalışarak tüm öğelerin bir arada kullanılması gerekir. Yukarıdaki öğelerden herhangi biri tek başına yeterli değildir. Süreçlerde çalışan insanların yaratıcılığı ile verileri, müşteri ve süreç anlayışını birleştirmeleri, Yalın Altı Sigma projelerindeki başarının temel şartıdır.

Altı Sigma iş performansının ve müşteri memnuniyet seviyesinin sürekli olarak artırılması için iş süreçlerinin iyileştirilmesi ve yeniden tasarlanması düşüncesine dayanan ve mükemmele yakın bir seviyeye ulaşmayı hedefleyen bir yaklaşımdır.

Altı Sigma felsefesi, meydana gelen hataların ve maliyetlerin azaltılması, iş süreçlerinin iyileştirilmesi, müşteri memnuniyet seviyesinin, firma prestijinin ve personel yetkinliğinin artması gibi birçok amacı içermektedir. Altı Sigma uygulama süreci neticesinde, olası sorun ve hatalar ortaya çıkarılmakta, bunlar düzeltilerek sürecin en kusursuz biçimini alması için gayret gösterilmektedir. Altı Sigma Modeli'ni uygulayan şirketlerde verimsizlik yaratan ve sigma seviyesinin düşmesine neden olan problemler mercek altına alınmaktadır. Altı Sigma Modeli, maliyetlerde ve hata oranında azalma, verim, pazar payı, müşteri ve çalışanların memnuniyetinde artış, kurum kültüründe olumlu değişim gibi konularda firmalara fayda sağlamaktadır. Altı Sigma uygulayan firmalar, ürün ve hizmetlerindeki hata oranını mümkün olan en düşük düzeye indirebilmektedir. Altı Sigma uygulayan bir firmanın ürettiği her bir milyon ürün ya da hizmette yalnızca 3,4 hata olasılığı vardır. Neredeyse sıfır hataya yaklaşan bu rakam, başarılı bir Altı Sigma uygulama süreci sonucunda elde edilebilir. Altı Sigma, kusur ve hataları en aza indirebilmek ve sıfır hataya yakın kalite düzeyini gerçekleştirebilmek için işletmelerin dikkatle uygulaması gereken, tüm dünyada geçerliliği defalarca kanıtlanmış bir yaklaşımdır [52].

3.1.2.5 Yaratıcı Problem Çözme Teorisi(TRIZ)

Mühendislerin yaratıcılığını, bilgisini ve problem çözme yeteneğini arttıracak ve çözüm noktasında imkansız görünen problemler konusunda atılım yapmalarını sağlayabilecek yeni bir teknik ve yöntemin adı olarak bilinmektedir. TRIZ, Rusça kökenli bir kelime olup "Teoriya Resheniya Izobreatatelskikh Zadatch" kelimelerinin baş harflerinden oluşmakta ve "Yaratıcı Problem Çözme Teorisi" anlamına gelmektedir. TRIZ teorisinin mantığı yeni bir problemin tasarımının önceden bilinen çözüm fikirleri üzerinden kurulması olarak düşünülür.

TRIZ yöntemi, 1946 yılında Sovyetler Birliği Patent Ofisi'nde çalışmakta olan Altshuller ve arkadaşları tarafından bulunmuş ve günümüze kadar gelerek geliştirilmiştir.

TRIZ yaratıcı mühendislige asagıda belirtilen yönlerde katkılar saglamıstır [53]:

- Teknolojik evrimin sistematik yapısını kesfederek alanlara baglı olmayan evrim eğilimlerini tanımlamıstır.

- Tasarım çözümleri için yeni sınıflandırma olusturmustur.
- Yaratıcı problemlerin temelinde çeliskilerin oldugunu ve bulusların bu çeliskileri yok ederek gerçekleştigini belirtmistir.
- Çelişkilerin ortadan kaldırılması için bazı temel prensipler önermiş ve bu prensiplere sistematik erişimi sağlamıştır.
- Sistemlerin veya tasarımların madde, alan ve bunların etkileşiminden oluşan modellerini kurarak ürünlerin fiziksel yapılarını dönüştürmek için genel-geçer kalıplar uygulamıştır.
- Tasarımcının veya mucidin psikolojik ataletini yenmek için bazı işlemler ortaya koymuştur.

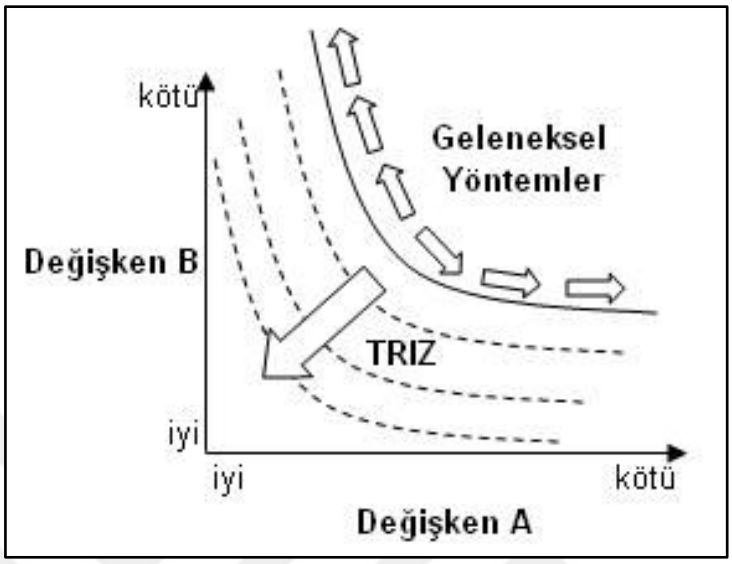
Yaratıcı bilgi birikiminin tamamına erişebilir ve bu birikime yeni bilgiler ekleyebilir. Mühendislik uygulamaları sırasında karşılaştığımız sorunların çözüm yöntemleri incelendiğinde bunların yaklaşık olarak %90' ının daha önceden düşünülerek kullanıldığı görülmüştür. Yöntemin kurucularından Altshuller bu bilgi üzerinden aşağıdaki gibi yaratıcılık ölçütünü tablolaştırmıştır.

Çizelge 3. 1 Yaratıcılık Seviyeleri [54]

Seviye	**Yaratıcılık Derecesi**	**Bütün Çözümler İçindeki % Değeri**	**Bilgi Kaynağı**
1	Bilinen Çözüm	% 32	Kişisel Bilgi
2	Küçük Yenilikler	% 45	Kurum İçi Bilgi
3	Büyük Yenilikler	% 18	Sektör İçi Bilgi
4	Yeni Kavram	% 4	Sektör Dışı Bilgi
5	Buluş	% 1	Tüm Bilgi

Çizelgeden anlaşılacağı üzere bir problemin çözümüne yönelik ortaya atılan fikirlerin %4'ü yeni bir kavram olarak nitelendirilmekteyken sadece %1'i yaratıcılık kapsamında bir buluş olarak nitelendirilmektedir [54].

Şekil 3.7'de TRIZ ile geleneksel yöntemler arasında, sistem değişkenlerinin iyileştirilmesi açısından yapılan karşılaştırma diyagramı yer almaktadır. Bu diyagramdan değişkenlerin iyileştirilmesinde TRIZ'i kullanmanın avantajı görülebilmektedir.



Şekil 3. 7 TRIZ ile Geleneksel Yöntemler Arasındaki Fark [53]

3.1.2.6 Veri Madenciliği Analitik Yaklaşımı

Veri madenciliği, çözümü zor olan iş problemlerini veriyi esas alarak analitik hipotez geliştirme ve istatistiksel metodojiler ile problemi analiz ederek çözüm önerisi sunmaktır. Süreç içerisinde datanın doğruluğu ve kurulan modellemeler önem arz etmektedir. Veri madenciliği, tarihsel süreç içerisindeki verileri kullanarak, ilerki süreçte alınacak aksiyonların tahmin edilmesini sağlamaktadır. Bu amaçla veri madenciliği için 3 teknoloji kullanılmaktadır.

- **Bilgi İşlem Gücü**

İşlenen verideki sunucuların işlemleri CPU larda yapılmakta ve CPU ların seri işlemler yapma kabileyiti yanında paralel işlemler için GPU yada üzerinde GPGPU programlama ile işlem hızı arttırılmıştır. Bu sayede data tutma problemleri aşılmakta ve günümüzde 'Big Data' ile bu yönde gelişmeler halen devam etmektedir.

- **Data Yönetimi**

Datanın iyi yönetimi datanın tutulması ile ilişkilidir. Günümüzde teknolojinin her alanında veriyi nicelikli halde tutmak önem arz etmekte ve datawarehouse sistemler olup olmadığına bakılmaksızın sistemli veri tutulması ileride ihtiyaç olunması durumunda hazır tutulmak istenmektedir.

- **Analitik Araçların Gelişmesi**

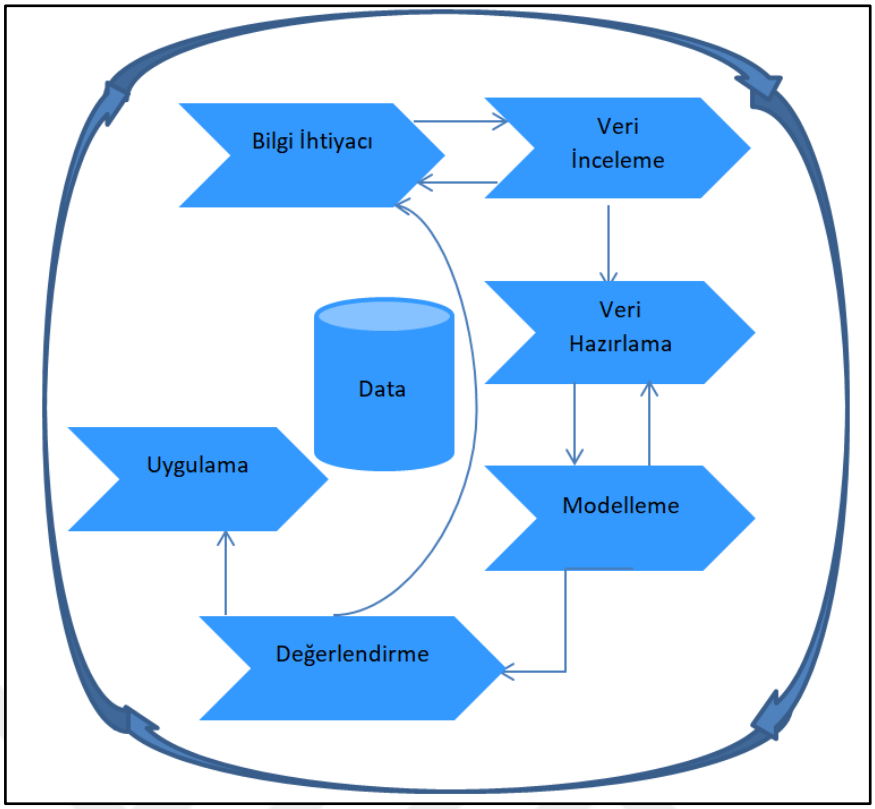
Veri madenciliği veri aracılığı ile problemleri çözmek adına anlamlı sonuç elde etmek zorundadır. Bu amaçla var olan yöntemleri geliştirmek ve teknikleri araştırmakla yükümlüdür.

Veri madenciliği gelişimini Raporlama, Veri Analizi ve Öngörü Üretme aşamaları ile özetleyebiliriz:

Raporlama: Amaç bilgi ekskliği ortadan kaldırmak olduğundan olası problemlerde verinin hızlı ve doğru şekilde alınması amacıyla birçok araç kullanılmaktadır. Excel, Otomatize Raporlama Araçları, Business Object, Hyperion bunlardan birkaçıdır.

Veri Analizi: İşleyiş sırasında belirsilikleri gidermek, analiz gerektiren probleme öngörü üretmek asıl amacımızdır. Veri yönetimine ek olarak analitik modellemeye de ihtiyaç bulunmaktadır. SPSS, SAS vb bunlara örnek gösterilebilir.

Öngörü Üretme: Analitik ekip tarafından problem çözmek esas amaçtır. Veri yönetimi, analitik modelleme yetkinlikleri ile kullanılan araçlar Rapid Miner, proje yönetim araçları, SAS örnek gösterilebilir.



Şekil 3. 8 Veri Madenciliği Süreci [55]

Şekil 3.8'de veri kaynakları veri inceleme adımında problemin tanımlanması yapılarak süreç başlanılır. Projenin yapılma amacının ve elde edilecek sonuçların başarı düzeylerinin nasıl ölçüleceği tanımlanmalıdır. Veri hazırlama adımında toplama, birleştirme ve temizleme, dönüştürme işlemleri yapılarak veri, modelleme adımına uygun hale getirilmektedir. Modelleme adımında, model kuruluş süreci denetimli ve denetimsiz öğrenimin kullanıldığı modellere göre farklılık göstermektedir. Denetimli öğrenmede sistemin amacı, verilen örneklerden hareket ederek her bir sınıfa ilişkin özelliklerin bulunmasıdır. Denetimsiz öğrenmede, ilgili örneklerin gözlenmesi ve bu örneklerin özellikleri arasındaki benzerliklerden hareket ederek sınıfların tanımlanması amaçlanır. Değerlendirme adımında kurulan modelin doğruluğu test edilir. Kullanma adımında, kurulan ve geçerliliği kabul edilen model doğrudan kullanılabilir veya bir başka uygulamanın alt parçası olabilir. Bütün bu adımlar gerçekleştirildikten sonra kurulan model izlenir [55].

Veri madenciliği projelerin hızlı, daha verimli ve daha az maliyetli gerçekleştirilmesi için kullanılan tekrarlı bir süreçtir.

3.1.3 Tasarım Deseni ve Avantajları

Tasarım deseni çalışan bir kod ya da algoritma olmayıp, problemin çözümünü anlaşılır bir şekilde ortaya koyulması için geliştirilen önerilerden oluşmaktadır. Gang of Four topluluğu tarafından yayınlanan tasarım desenleri, Nesne Yönelimli Programlama da karşılaşılmış tasarım sorunlarına üretilmiş optimum çözümler olarak tanımlanabilir. Tasarım desenleri bizlere daha yönetilebilir ve okunabilir kod yazmak konusunda avantajlar sağlar. Bunun yanında performans ve geliştirilebilirlik konusunda da katkıları vardır. Gerçek hayatta karşılaştığımız problemlerin tasarım yöntemleri ile ifade edilmesi, yazılım geliştirme aşamasında sınıf (class) kavramının iyi bir şekilde tasarlanmasına yardımcı olmaktadır. Sınıf kavramının iyi oluşturulması, nesne ile sınıf arasındaki ilişkinin daha iyi açıklanmasına da imkan vermektedir. Tasarım yöntemlerinin uygulanması programın daha kolay bir yapıda kurgulanmasına, daha kısa sürede geliştirme yapılmasına ve uygulamanın kalitesini arttırmasına olanak sağlamaktadır [56].

Tasarım deseni oluşturmanın sisteme sağlayacağı en büyük fayda, olası sistem değişikliklerine daha hızlı adapte olmak, kodların tekrar kullanılabilir olmasını yani sistemin çalışmasını etkilemeden yazılıma esneklik kazandırmaktır. Bu fayda çevik bir mimari yöntemin gerekliliği olup, günümüz kullanıcısına elle tutulur ve tatmin edici bir istem sunar.

Günümüzde analist ya da sistem geliştiricilerin çeşitli iş analizi teknikleri kullanarak problemi programcıya doğru ifade etmek için tasarım desenleri oluşturulur. Nesneye yönelik programlamanın bu modellerin gelişiminde faydası olmakla birlikte; eksik kaldığı durumlarda tasarım desenleri kullanılmaktadır. Bu çalışmada da iş analitiğinin temelini oluşturulan tasarım desenleri kullanımına yer verilerek programlama sürecine katkısı ve dolayısıyla kullanıcının sistemden sağladığı fayda gösterilmeye çalışılacaktır.

3.1.4 Yazılım Geliştirme Metodolojilerini Kullanma

Yazılım sektörü, teknolojiyi esas alan bir sektördür. Bu nedenle sürekli kendisini geliştirmeli ve çağın gerisinde kalmamalıdır. Yeni teknik çözümler üretirken ileri teknoloji bilgilerini kullanmalıdır. Bu nedenle yazılım firmalarının teknolojik kapasiteleri ve kaynakları, ürün geliştirme süreci için kritik önem taşımaktadır. Yazılım firmalarının varlıklarını devam ettirebilmeleri, sürekli ve sorunsuz ürünler yaratmalarına ve bu ürünleri müşteriye teslim etme sürelerine bağlıdır.

Yazılım ürünleri geliştirilirken müşteri ihtiyaçlarına odaklanılmalı, yapılacak işin teknik üstünlüğüne önem verilmelidir.

Yazılım geliştirme süreci, "İş gereksinimlerinin yazılım ürünlerine dönüşmesi için yapılması gerekli olan faaliyetlerin tanımlanması ve bu gereksinimlerin bir yazılım ürününe dönüştürülmesi" olarak tanımlanmaktadır [57],[58].

Yazılım geliştirme kendini tekrar eden bir süreç değildir. Ürün geliştirme ekibindeki her bir bireyin başarısı ürünün kalitesini etkilemektedir. Yazılım geliştirme sürecinin bir aşamasındaki başarısızlık, önceki adımlardaki başarısızlığın bir sonucudur. Örneğin, analiz aşaması iyi bir şekilde tamamlanmamışsa, geliştirme aşamasında başarısızlıkların ortaya çıkması kaçınılmazdır. Bu nedenle yazılım projeleri; işin tanımlanması, analizlerin yapılması, programın tasarımının yapılması, yazılımın kodlanması, test edilmesi ve dokümanların hazırlanması aşamalarının tümünü kapsamaktadır.

Yazılım Süreçlerinin Genel Adımları:

- Çözümleme: Yazılım sürecine konu olan işi anlayabilmek için parçalarına ayırmak anlamına gelmektedir. Bu aşamada kullanıcı gereksinimleri ile beraber program parçaları arasındaki ilişkiler ve etkileşimler(nesne ve sınıflar) sistemi anlamaya yönelik çalışmalar yapılmaktadır.
- Tasarım: Geliştirme sürecinin her devresinde izlenecek yol ve işlemleri tasarlamak amaçlı kullanılan UML şemalarının kullanımı ile çözümleme aşamasındaki sorun çözüme kavuşturulur.

- Gerçekleme: Elde edilen tasarımın herhenabi bir programlama dile ile kodlanmasıdır.
- Sınama: Yazılım süreci ilerledikçe isteklerin belirlenememesi veya tasarım aşamasındaki sorunlardan dolayı hataların artması ihtimaline karşı sık sık sınama yapılması gerekmektedir.
- Bakım: Yazılım kullanılmaya başlaması ile kodlama hatalarının, tasarım hatalarının veya gereksinim/analiz hatalarının düzeltilmesi gibi sistemde yapılan her türlü değişiklik bu kapsamın konusu olmaktadır.

Yazılım Projelerinin Başarısız Olmasının Nedenleri:

Yazılım projelerinin başarısız olmasının sebeplerini inceleyecek olursak şu şekilde sıralayabiliriz [59]:

- İsteklerin doğru ifade edilememesi ve zamanla değişim göstermesi

Yazılım projeleri, bir anlamda hayal edilen işin yapılması gerçeğe dönüştürülmesidir. Bu nedenle projenin herhangi bir aşamasında, düşünülmeyen, açık kalan bir konunun olması projenin süresinin uzamasına neden olabilir.

Yazılımı talep eden kişi veya kurumlar ne istediklerini tam olarak bilemeyebilirler ya da doğru ifade edemeyebilirler. Ayrıca zamanla değişen ihtiyaçlar sebebiyle müşteri gereksinimleri ve istekleri farklılaşabilmektedir. Müşteri isteklerinin tam ve net elde edilememesi veya başlangıçta belirtilmeyen yeni eklemelerin proje sonuna doğru istenmesi sebebiyle projeler başarısızlığa uğramaktadır. Bu gibi durumlarda ciddi zaman ve parasal kayıplar gözlemlenmektedir.

- İstekleri dile getiren ekibin zamanla değişmesi

Yazılım projeleri genellikle uzun soluklu projelerdir. Projenin analiz aşamasında ürün isteğinde bulunan ekipte, başka projeye geçiş - terfi - iş değişikliği gibi nedenlerle değişiklikler olabilir. Bu gibi durumlarda projeden uzaklaşan kişinin, işi doğru bir şekilde diğer kullanıcıya devretmesi önem kazanmaktadır. İşi devralan kişinin, gelişmelerden ve yeniliklerden haberdar olması ve bilgi eksikliğinin olmaması gerekmektedir.

- Gerçekçi olmayan zaman tahminleri

Gerçekçi olmayan zaman tahminleri projelerin uzamasında bir diğer etkendir. Müşteriler işi en kısa zamanda teslim almak istedikleri için, projenin zamanından önce teslim edilmesi için baskı uygularlar. Bu baskı karşısında işi kaybetmek istemeyen yöneticiler proje planlamasına daha az zaman ayırarak gerçekçi olmayan zaman tahminleri sunabilmektedir. Bu gibi durumlar profesyonel yönetici ile çalışıldığında aşılabilmektedir.

- Proje ekibinin doğru şekilde oluşturulamaması

Yazılım geliştirme projesine ait uzman bir ekip oluşturulamadığında ve proje yöneticilerinin projeye gereken ilgiyi göstermemesi durumlarında yazılım projelerinin başarısız olması kaçınılmazdır. Proje yöneticisinin, yazılım geliştirme konusunda bilgi sahibi olan bir kişi olması gerekmektedir. Aynı zamanda, proje ekibi içerisindeki her bireyin, sorumluluklarını tam olarak üstlenmesi ve takım ruhuna uygun bir şekilde çalışılması gerekmektedir.

- Müşteri ile iletişimde kopuklukların yaşanması

Proje yönetiminin müşteriler ile olan iletişimsizliği de projeleri başarısızlık riski ile yüz yüze getirebilmektedir. Bu nedenle müşterinin isteklerini doğru anlatamasa bile soracağı sorular ile müşterinin aklındaki projenin tam olarak anlaşılması sağlanmalıdır. Açık kalan her konu için anında müşteri ile iletişime geçilerek doğru cevabın alınması ve projeye bu şekilde devam edilmesi gerekmektedir.

- Geliştirmenin test edildiği ortam ile yazılımın kullanılacağı gerçek ortam arasında farklılıkların olması

Yazılım ekibi geliştirmeyi test ortamda yaparak, kullanıcı onayı alındıktan sonra gerçek ortama taşır. Geliştirmenin yapıldığı test ortam, gerçek ortama en yakın olan ortam olmalıdır. Bu şekilde, projenin gerçek ortamdaki uygulamalarla olan uyumu test edilebilir.

Aksi durumda, gerçek ortamdaki uygulamalar ile çakışmalar yaşanabilir ve test ortamda yaşanmayan sorunlar gerçek ortamda karşımıza çıkabilir. Yaşanacak olan bu sorunlar, projenin başarısını etkilemektedir.

- Kullanıcı eğitimlerinin verilmemesi

Geliştirilen yazılım sonrasında, kullanıcıların bilgi eksikliklerini tamamlamak için, kullanıcılara eğitimler verilmelidir. Aksi halde, proje tamamlandıktan sonra kullanıcılardan yanlış geribildirimler alınabilir. Örneğin, gerçekte olması gereken bir uygulama için kullanıcı "olmamalıydı" diyerek, projenin yeniden incelenmesine sebep olabilir. Bu nedenle, kullanıcılara eğitim verilerek; detaylar onlarla da paylaşılmalıdır.

Yazılım projelerinde, zamanla değişen yapıya ayak uydurmak gerekmektedir. Projelerin başarısızlığının bir diğer sebebi de bu uyumu yakalayamamaktır. Zamanla değişen bu yapıya bir de yukarıda bahsettiğim maddeler de eklendiğinde; sorun daha da çözülemez hale gelebilmektedir. Bu nedenle;

- Geliştirme süreci boyunca kullanıcıların değişmemesi,
- Kullanıcı ile tasarımcı arasında etkili ve iyi bir iletişimin olması,
- Çalışılan test ortamının, gerçek ortama çok yakın bir ortam olması,
- Kullanıcıların bilgi eksikliklerinin giderilmesi,
- Gereksinimlerin doğru belirlenmesi

Hatasız yazılımların ortaya çıkmasını, yazılım kalitesinin artmasını sağlamaktadır. Bu durum, yazılım firmasına da avantaj sağlamaktadır. Başarılı bir proje hem müşteri memnuniyetini hem de piyasada firmaya olan güveni arttırmaktadır.

3.1.5 Şelale - Waterfal Yazılım Geliştirme

Gereksinim Analizi →Analiz→Geliştirme→Test→Kullanıcı Kabul ve canlı ortama alma

- Şelalenin her basamağında yer alan aktiviteler eksiksiz olarak yerine getirilir. Bu, sonraki basamağa geçmenin şartıdır.

- Her safhanın sonunda bir doküman oluşturulur. Bu yüzden waterfall modeli doküman güdümlüdür.
- Kullanıcı katılımı başlangıç safhasında gerçekleşir. Kullanıcı gereksinimleri bu safhada tespit edilir ve detaylandırılır.

Geleneksel yöntemin dezavantajları:

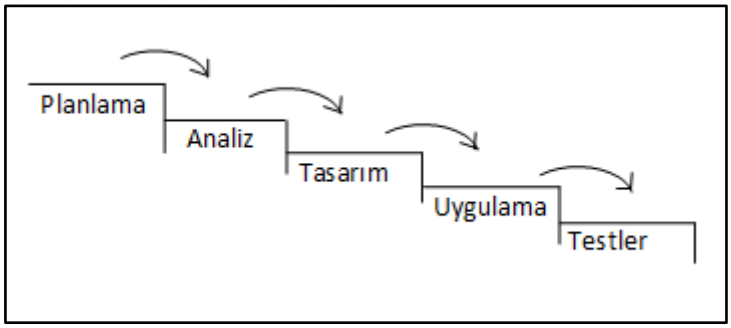
Nihai mükemmel ürün tüm özellikleriyle geliştirilerek kullanıma alınır, müşteri memnuniyeti belirsizdir ve proje tamamlandıktan sonra anlaşılır.

•Nihai mükemmel ürüne göre kaynak planlaması yapılır.

•Ürün sorumlusu projenin başında ve kabul aşamasında rol alır, öncelik yönetimi proje başında yapılır, değişiklik yönetimi ise ek talep ve kaynak planlaması ile yapılabilir.

•Ürün sorumlusu ve proje paydaşlar sadece proje başında hizalanır ve proje sonunda gerçekleşmeler değerlendirilir.

•Proje zarfınca yönetilemeyen riskler ve maliyetler sıklıkla gerçekleşebilir.



Şekil 3. 9 Geleneksel süreçlerin lineer yapısı

Geleneksel Yöntemlerin Yazılım Projelerinde Başarısız Olmasının Nedenleri:

Yazılım geliştirme süreci tamamlandıktan sonra, test edilmek üzere kullanıcılara iletilir. Burada yapılan işin, kullanıcıların isteklerini tam olarak karşılamadığının tespit edilmesi halinde düzeltilmesi oldukça zaman almaktadır. Geleneksel yöntemlere göre, istenilen değişikliklerin yapılabilmesi, yazılım sürecinin yeniden başlaması demektir.

Süreç analiz aşamasına yeniden geri dönerek, şemalar yeniden hazırlanır ve yazılım ekibine yeniden iletilir.

Yazılım projeleri, uzun soluklu projeler olması sebebiyle, proje üzerinde çalışılırken bile isteklerde değişiklikler talep edilebilir. Bu durum da şemaların revize edilmesini gerektirmektedir. Mevcut dökümanlar yeniden güncellenerek, yazılımcıya yeniden iletilir. Bu durum zaman, emek ve para kaybına neden olmaktadır.

Bu nedenle teknoloji ile iç içe çalışan yazılım projelerinin esnekliğe ihtiyacı vardır. Değişen yapıya hemen ayak uydurabilmesi için de, projelerin geleneksel yöntemlerden uzaklaşması gerekmektedir.

3.1.6 Çevik - Agile Yazılım Geliştirme

Sürekli değişen çevreye uyum sağlayabilme becerisi olarak tanımlanmaktadır. Gelişen teknoloji ile beraber değişime ayak uyduramayan projelerin iptal edilmesi veya ihtiyacın ortaya çıkmasıyla teslim edilme süresi arasındaki farkın çok uzun süre sonra gerçekleşmesi uygulama geliştirme aşamasında gecikme yaşanmasına neden olmaktadır. Yaşanılan sıkıntılar konusunda gereksinimleri karşılayabilmek adına 2001 yılında Agile Manifestonun temelleri oluşturulmuştur.

Agile; süreçler ve araçlar yerine bireyler ve etkileşimlere, detaylı dokümanlar yerine çalışan yazılıma, sözleşme pazarlığı yerine müşteri memnuniyetine, bir plana bağlı kalma yerine değişikliğe ayak uydurabilmeyi daha değerli bulmaktadır.

Agile ile yazılım geliştirmenin özellikleri ve avantajları:

- Analiz, tasarım, yazılım ve test sürekli aktivitelerdir.
- Yazılım geliştirme tekrarlıdır.
- Planlama duruma göre uyumlanır.
- Roller keskin çizgilerle ayrılmamıştır.
- İstekler değişebilir.
- Çalışan yazılım başarının ilk ölçüsüdür.
- Kapsam değişebilir.

- En yalın ürünün kısa sürede kullanıma alınmasını sağlar ve hızlıca müşteri memnuniyeti yaratır.
- Pazarda sürekli değişen ihtiyaçlar doğrultusunda etkin kaynak yönetimi sağlar.
- Ürün sorumlusu üretim sürecinin içinde sürekli rol alır, değişiklik ve öncelik yönetimi üretimin her aşamasında yapılabilir.
- Ürün sorumlusu ve proje ekibi gerçekleşmeler ve hedefler doğrultusunda sürekli hizalanır.
- Üretkenlik ve kalite artışı sağlar, maliyetler düşer ve riskler azalır.

3.1.6.1 Yazılım Projelerinde Çevik Metodların Tercih Edilmesi

Yazılım projelerinde müşterilerin gereksinimlerinin her zaman net olmadığından, zamana bağlı olarak değişiklik gösterebildiğinden daha önce bahsetmiştim. Bu şekilde net olarak ifade edilemeyen projelerde çevik yöntemlerin kullanılması, müşteri gereksinimini daha iyi anlamamızı sağlar.

Çevik yöntemlerde, proje süreçleri esnektir. Proje başlangıç aşamasında önemli görülen bir ihtiyacın, ilerleyen aşamalarda önemsiz olduğunun anlaşılması durumunda; bu ihtiyacın projeden çıkarılması ve ihtiyacın önceki ve sonraki adımlarının iş akışı üzerinde yeniden tasarlanması geleneksel yöntemlere göre çok daha hızlıdır. Geleneksel yöntemlerde bu gibi değişiklikler için, iki farklı yöntem izlenebilmektedir. Ya projenin tamamlanması beklenmek zorundadır.

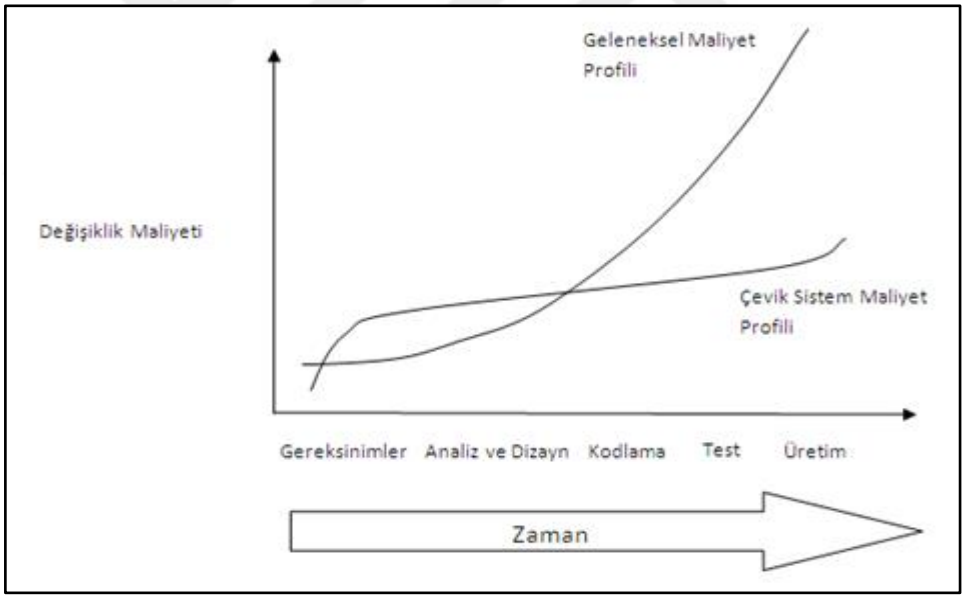
Sonrasında yeniden bir analiz yapılarak istenmeyen ihtiyaç projeden çıkarılarak yeni bir geliştirme yapılacaktır. Ya da projede gelinen noktadan en başa dönülerek yeniden analiz yapılarak ihtiyaçlar yeniden belirlenecektir. Her iki durumda çalışmalar kullanılamaz hale gelmekte ve çalışılan zaman boşa geçen zaman olarak gözükmektedir. Oysa ki, çevik metodolojilerde, proje içerisinden istenmeyen bir iş parçacığını çıkartmak ya da yeni bir iş parçacığını projeye eklemek çok daha kolaydır.

Kanban metodolojisi için bir iterasyon zamanı kadar beklemek ya da scrum için bir sprint süresi kadar beklemek; istenilen değişikliği uygulamaya almak için yeterli olmaktadır.

Çevik yöntemlerin başlıca özelliği, veri modelleri ve ara yüzü modelleri gibi modelleme tekniklerinin neler olduğunu ve bunların ayrıntılarını söylemek yerine bu tekniklerin nasıl uygulanması gerektiğini söylemesidir. Örneğin; yapılan projelerin test edilmesi gerektiğini belirtirken, nasıl test edileceğine değinmemektedir.

Çevik yöntemler bir projenin etkili, hızlı bir şekilde ortaya çıkarılmasında, müşterinin ihtiyaçlarını karşılamasında ve aynı zamanda da her türlü değişikliğe kolayca adapte olabilmesinde geliştiricilere yol gösterir.

Projenin parçalarının önce tasarlanıp ardından hemen geliştirilmesi ve önceden ne yapılacağını, detaylı yol haritasını ve tasarımını tahmin etmek çok güçtür. Bu nedenle projeyi küçük parçalara ayırarak bu tahminlerdeki sapmaları en aza indirmek çevik yöntemler sayesinde mümkündür. Böylece küçük parçalara ayrılmış olan işlerin analiz, tasarım ve test etme süreçlerinin ne kadar zaman alacağını tahmin etmek daha kolay olacaktır.



Şekil 3. 10 Değişiklik maliyeti-zaman grafiği [60]

3.1.7 Scrum

Yeni bir ürün prototipi veya varolan bir sistemin yönetim, geliştirme ve bakım medolojisi anlamına gelmektedir. Scrum çevik yazılım geliştirme yaklaşımlarını benimser. Gereksinimleri sağlayacak, başarılı ve kaliteli bir yazılım üretimi için bir araya gelmiş takım ve müşteri birlikteliği Scrum'ın getirdiği en büyük avantajlardandır.

Scrum içerisinde temel prensip düzenli geri bildirimde bulunmak ve kısa zamanda hedefe ulaşmaktır. Ürün sahibi, scrum master ve takım olarak üç ana rol üzerinden scrum yürütülür.

Ürün sahibi sorumlulukları;

- Öncelikleri önceliklendirme ve yönetme
- Çalışılan vizyonun paylaşımcı olması
- Her yinelemede ürün kabülü yapılması
- Gereksinimleri toplama
- Projenin yatırımının geri dönüşümünden sorumlu olma
- Dağıtım planını yönetme

Scrum içerisinde master sorumlulukları;

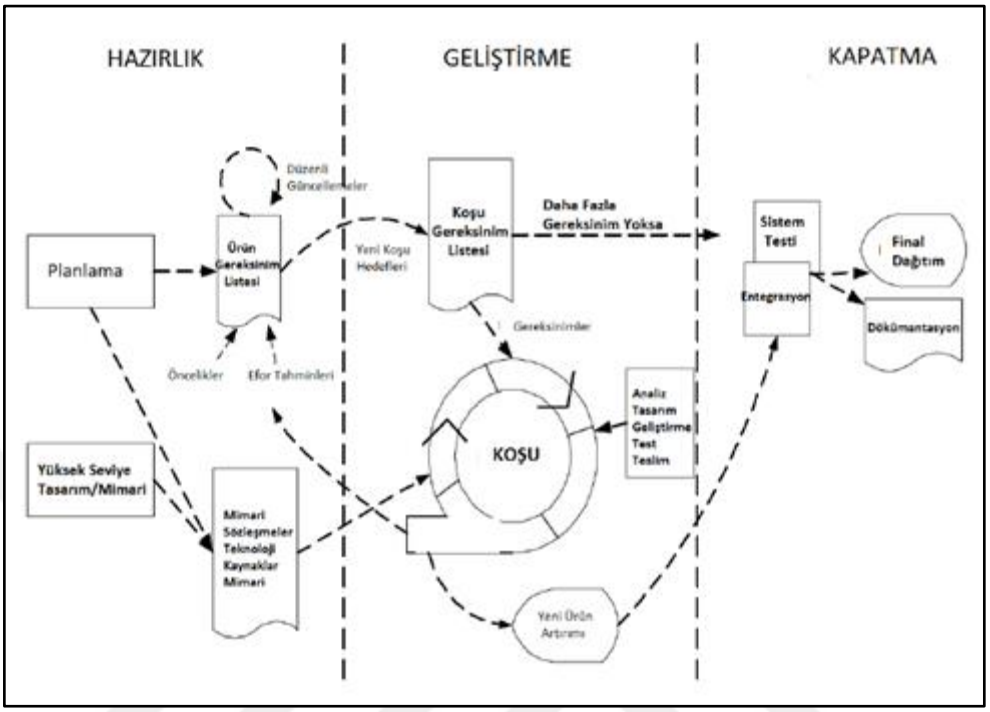
- Tüm fonksiyon ve roller arasında yakın iş birliği sağlanması
- Sürecin doğru işletilmesi
- Takımı baskılardan ve dış etkilerden koruması
- Takımın üretken olmasını ve yönlendirilmesini sağlama
- Engelleri kaldırma

Scrum içerisinde takımların özellikleri;

- Küçük takımlardan oluşur (7+-2)
- Takım üyeleri kendilerini organize eder
- Gereksinimler doğrultusunda üre tahminini yapar
- Hedefi seçerek çalışma sonucunu belirlerler

- Çalışma çıktılarını ürün sahibine sunar ve görüşlerini alırlar

- Proje sınırları dahilinde istediklerini yapmakta serbesttirler



Şekil 3. 11 Scrum'da süreçlerin genel görünümü [61]

3.1.8 En Yalın Ürün Kavramı (Mvp - Minimum Viable Product)

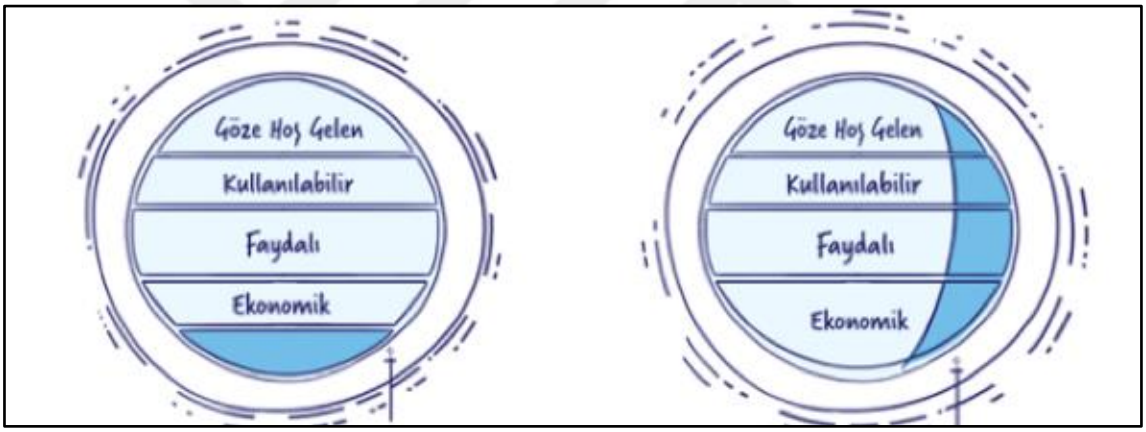
Müşterinin/kullanıcının ihtiyaç duyduğu minimum fonksiyoneliteyi sağlayan en basit, yalın ve çalışan üründür. Yani en yalın ürünü hipotezinizi test etmek için yaptığınız bir deney olarak düşünülmelidir.Ürüne olan talebi ölçmek istiyorsanız basit bir sayfa yapıp ön sipariş almak yalın ürününüz olabilir. Eğer bir çok detayını hayal ettiğiniz bir ürün varsa yapmanız gereken, en çok talep göreceğini düşündüğünüz iki veya üç özellik için ayrı web sayfaları hazırlayıp sonra hepsine çok küçük bütçelerle aynı miktarda reklam verip sonra hangisine daha çok ilgi olduğunu ölçmektir.

En temel özelliği keşfedip yalın ürününüzü ortaya çıkardıktan sonra ise çeşitli geri bildirim araçlarıyla bir çok özelliği müşterilerinize oylattırıp en çok oy alanları ürününüze ekleyebilirsiniz, hatta hiç oy almayan özelliklerinizi de komple yapmaktan vazgeçebilirsiniz [62].

Faydaları:

- Ürünün erkenden kullanıcıya sunumu ve kullanıma alınması imkanı sağlar.
- Geliştirme maliyetini en aza indirir.
- Iterative bir şekilde sürümler yapmaya olanak sağlar.
- Geri bildirimler ve hatalardan ders çıkarılmasını sağlar.

En yalın ürünle ilgili en çok yapılan hata en yalın ürünün basit, bir çok özelliği henüz tamamlanmamış maket gibi hayal edilmesidir. En yalın ürün fonksiyonel olarak düzgün çalışmak zorundadır ve kötü bir görünümü de olmak zorunda değildir. En yalın ürününüzün tam çalışmaması markanıza zarar verebileceği gibi güven vermezse gerçek müşteriye dönüştürme oranlarınızı da kötü etkiler ve deneyinizin sonuçlarını da olduğundan kötü gösterebilir. Aşağıdaki şekil 3.12'de En Yalın Ürünün nasıl olması gerektiği ile ilgili ipuçları verir:



Şekil 3. 12 En Yalın Ürün Kavramı Nasıl Olmamalı/Olmalı [62]

Henrik Kniberg'e göre yalın ürün; çalışan, problem çözen ve size bir hipotezinizi kanıtlamaya yarayan çözümdür [63].

3.1.9 Birleşik Modelleme Dili

"Unified Modelling Language" 'in kısaltması olan UML, Türkçe anlam karşılığı olarak "Birleşik Modelleme Dili" olarak çevrilebilir. UML bir programlama dili olarak değil, karşımıza ele alınacak işin metodolojisinin nasıl modellenebileceğini ifade eden ve yazılım projelerinin tanımlanması ve tasarlanması amacıyla geliştirilmiş bir modelleme dilidir.

Projelerimizde problemlerimizi parçalara bölebiliyorsak ve bu parçalar arasında ilişki kurabiliyorsak UML modelleme yapısını kullanabiliriz demektir. UML, nesneye yönelik programlama dillerinde kullanılan tüm diyagramları kapsayarak standartlaştırmaktadır.

UML modellemenin iş analizi teknikleri arasında kullanmanın faydaları:

- UML, analiz ve tasarımı detaylandırmamız için kullanıldığından kodlama aşamasında programdan bekleneni ve programın ilerde nasıl yürütülebileceği konusunda bilgi verir.
- Programın çalışmasında oluşabilecek mantıksal hataların en aza indirgenmesinde rol oynar.
- Kodları yeniliklere karşı daha kolay revize etmeyi ve tekrar kullanılabilen kodların sayısını artırarak hem karlılığı artıracak hem de maliyetin düşmesini sağlayacaktır.
- Kolay ve anlaşılır olması ortak projelerde iletişimi olumlu yönde etkileyecektir.

UML Diyagramları yapısal diyagramlar, davranış diyagramları ve etkileşim diyagramları olarak 3 grupta incelenir:

Yapısal Diyagramlar:

- Nesne(Object) diyagramı
- Birleşik Yapısal(Composite Structure) diyagramı
- Sınıf(Class) diyagramı
- Yerleşim(Deployment) diyagramı
- Bileşen(Component
- Paket(Packet) diyagramı

Davranış Diyagramları:

- Kullanıcı senaryosu(User case)diyagramı

- Durum şeması(Statechart) diyagram
- İşbirliği(Collaboration)diyagram

Etkileşim Diyagramları:

- Ardışıklık/Etkileşim(Sequnce)diyagramı
- Etkileşim tanıtma(Interaction overview) diyagramı
- Haberleşme(Communication) diyagramı
- Zamanlama(Timing)diyagramı

UML nesneye dayalı programlama dilleri için uygun modelleme dilidir. Tasarım aşamasının yazılım aşamasına geçmeden önce yapılması, yapılacak hata maliyetlerini düşürmede büyük etkendir. Yazılım tasarımının başında oluşturulacak sistemin net bir resmini bir bütün olarak görebilmeyi ve ilgili bileşenler arasındaki ilişkiyi kavrayarak, yazılım tasarım hatalarını azaltmaktadır [64].

3.2 Uygulama Alanının Çözümlenmesi (Domaın Analysıs)

Amaç, elde edilen verilerin analiz modeline taşınması ve uygulama alanını anlamaktır. Bu tez kapsamında kredi skorlama sisteminde kullanılan modellemeler uygulamalı olarak gösterilerek karar vericinin süreç içerisinde yapacağı değişikliklere daha hızlı ve doğru hamle ile ilerleyebileceği gösterilecektir.

3.2.1 Use Case Diagram (Kullanım Senaryosu)

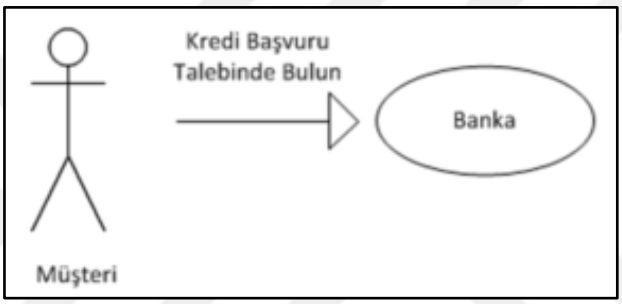
“Use Case” Türkçe anlam karşılığı olarak “Kullanım Senaryosu” olarak karşımıza çıkmaktadır. Kullanmadaki faydamız; kullanıcı açısından projenin amacını ortaya koymaktır.

Aktörler(Kullanıcı Grupları) projede amacımıza hizmet eden sistemin temel aktörlerinin belirlenmesi ve rollerinin tanımının yapılması gerekmektedir. Kullanıcıların sistem içerisindeki rolleri; müşteri yada kullanıcıların sistemden sağlayacağı faydaları ortaya koymaktadır.

Bu sayede kullanıcı perspektifinden bakıldığında sistem analizinde yazılımın gereksinimleri elde edilmektedir. Çalışmamızda ele alınacak aktörler;

- Müşteri
- Banka
- Alt Sistem

Bu aktörlerin sistemle etkileşimlerinde amaçlar farklı olabilir. Ancak her aktörün aynı anda tek bir amacı olduğu varsayımı üzerinden gidilecektir. Kullanım durum senaryoları bize sistemin gereksinimlerinin belirlenmesinde yardımcı olmaktadır. Çalışmamızda sistemimizin temel kullanıcısı müşteridir.

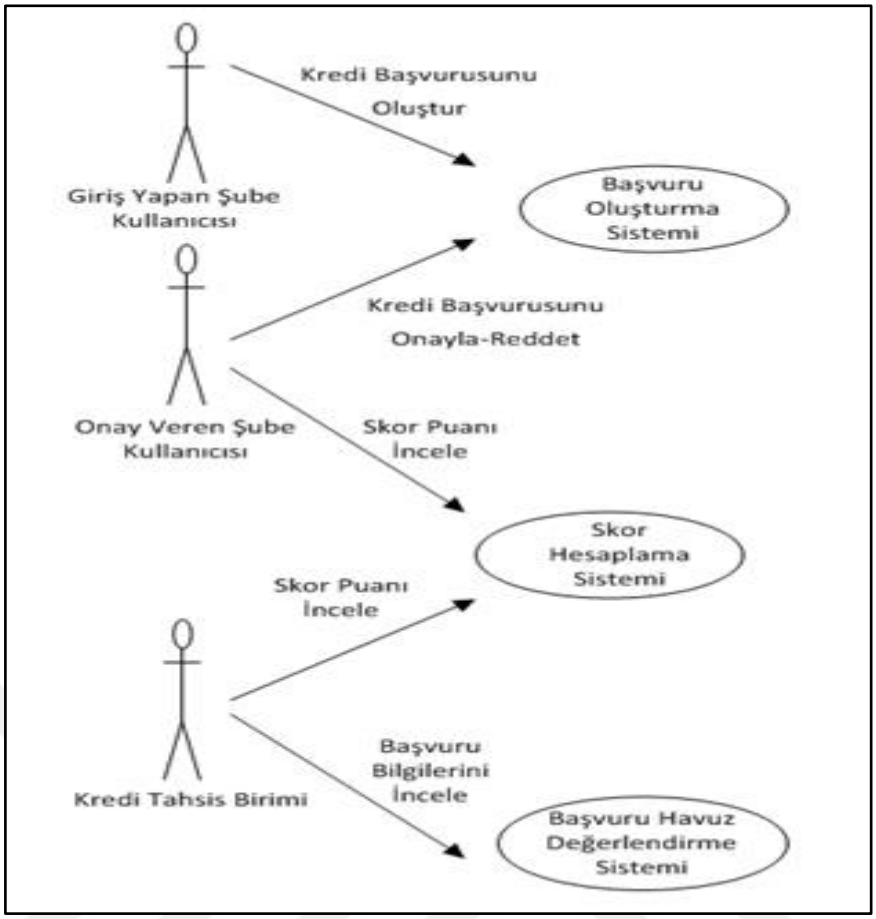


Şekil 3. 13 Müşteri Aktörünün Sistem İle İlişkisi

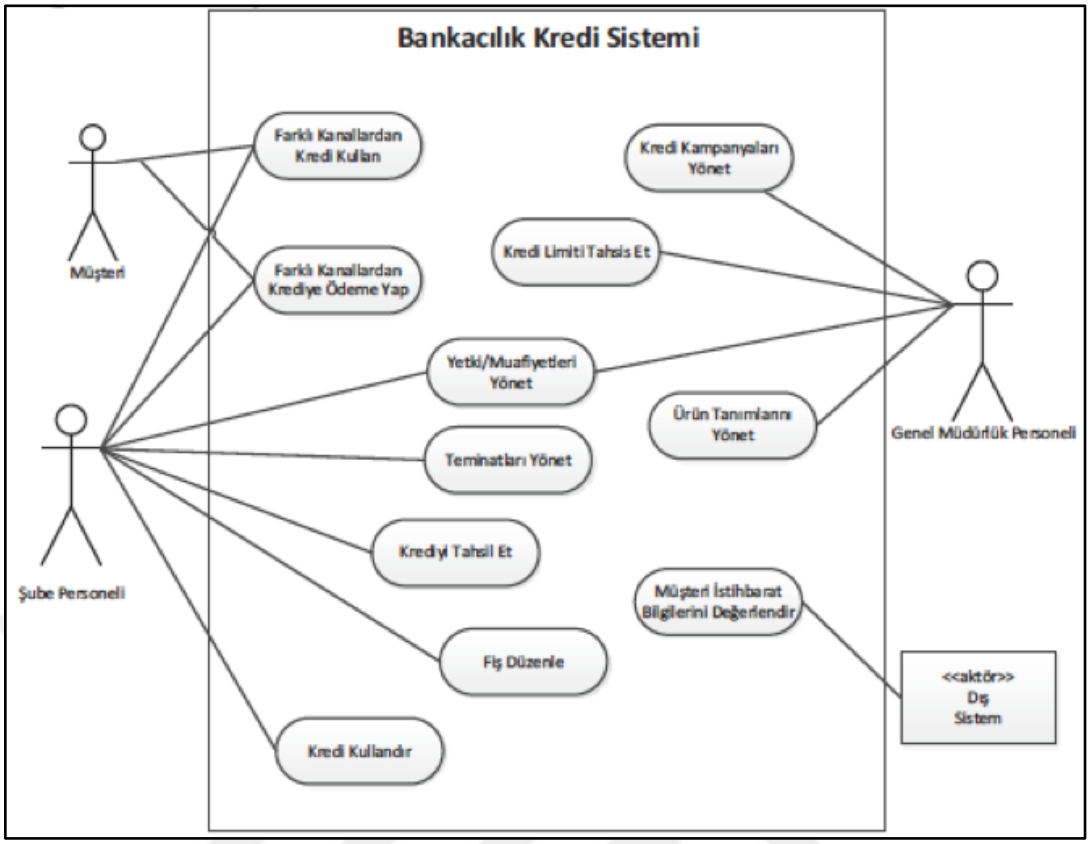
Diyagramın esas amacı gereksinimleri belirlemek ve olabildiğince temel anlatımla basite indirgemektir. Çalışma kapsamında Skor kart puanının kredinin onay/red sürecindeki etkisini kullanıcı açısından incelediğimizde aşağıdaki diyagramını oluşturabiliriz. Sistemden beklentimiz nelerdir; farklı aktörlerin sisteme bakış açısı ele alınır. Böylelikle sistemi çok rahat karşımızdaki kullanıcılara anlatabiliriz.

Bu diyagramda amaç kullanıcı açısından kredi sürecinde skor kart puanının etkisi, sistemin nasıl çalıştığını kullanıcı gözünden görmemizi sağlar.

Skor Hesaplama fonksiyonunun içinde mutlaka kredi başvurusu olması gerekmektedir bu açıdan diyagramımızda içerme-includes olarak tanımlıdır. Başvuru havuz değerlendirme sistemi ise diyagramımızda onay aşamasında skorun düşük olması durumunda geçerli olup genişletme- extend olarak tanımlanmaktadır.



Şekil 3. 14 Banka Kullanıcı Aktörlerinin Sistem İle İlişkisi



Şekil 3. 15 Kullanım Senaryosu Örneği

Use Case'lerin işlevinin üç temel özelliği vardır[65] :

- Sistemin bütününü görebilmek: Sistemin bütününü görebilmek: Sunulacak hizmetin sistem içinde nasıl gerçekleştirileceğini anlamak,
- Sistemin sınırlarını belirlemek: Tasarımlanan sistemin dışarıdan nasıl görüleceğini ve programcılara yol gösterebilecek şekilde belirlenmesi,
- Test için Referans Oluşturmak: Sistem oluştukça eklenen yeni unsurları kaldırıp kaldıramayacağını anlamak.

Use Case yaklaşımıyla yeni tanışan pek çok kişiye use case'lerin bulunmaları mistik ve doğruluğu çok zor test edilebilir gelebilir. Bunun nedeni geleneksel düşünce şekillerinin referanslarını değişmek zorunda olmalarıdır. Ancak yeni referansa alıştıkça aslında use case bulmanın pek güç olmadığını ve kişinin use case modelinin eksiksizliğinin çok net olarak ölçülebileceğini görürler.

Yapılması gereken kişinin kensini Aktör yerine koyması ve kendisi için gözlenebilen ve açık bir şekilde bir fayda ifade eden fiillerle sistemin temel bir özelliğini ifade etmesidir.

3.2.2 Veri Akış Diagramı(DFD-Data Flow Diagram)

Organizasyonun topolojisinin iş fonksiyonlarındaki süreçlerini detaylı bir biçimde analiz edilmesi ve yeni süreçlerin tasarlanması için kullanılmaktadır[66]. Sistemi çözen analistin veriyi takip etme süreci olarak tanımlanmaktadır. Bilginin nerelerden geçtiği ve sistemi modellerken hangi aşamalardan geçtiğini anlamamıza yardımcı olmaktadır.

'Veri Akışı' kavramı ilk duyduğumuzdaki anlamı ile, sistemde hareket eden veriyi ifade etmektedir. Her bir veri akışı ok sembolü ile ifade edilen bilgiler ile tanımlanır. Bilgi tek bir veri içerdiği gibi birden fazla veriyi de ifade edebilir.

Veri Akış Diyagramında Kullanılan Sembol ve Kurallar:

Proses(İşlem-Süreç): Yazılım literatüründe yapılan bir fonksiyonu yani aktiviteyi tanımlamak için kullanılırlar. Prosesler numara ve isimle gösterilip, faaliyeti anlaşılır bir emir cümlesi ile tanımlamalıdır. Simge olarak yuvarlatılmış kutu ile temsil edilir.

Veri Akışı: Süreçler, veri depoları ve dışsal birimler arasındaki veri hareketini ifade eder. Veri akışı oklar ile gösterilip, okun yönü veri akış yönünü göstermektedir.

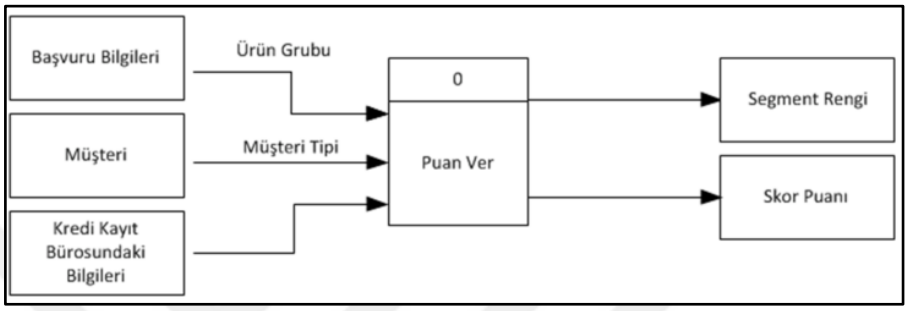
Dışsal Birim: Sistemden fayda alan kişi veya hedef olarak tanımlanabilir. Simgesi kare ile ifade edilir.

Veri Deposu: Verinin saklandığı yerdir, 'veri nerede' sorusuna cevap vermektedir (hardisk, dosya dolabı vs.) Simge olarak ucu açık dikdörtgen ile temsil edilir.

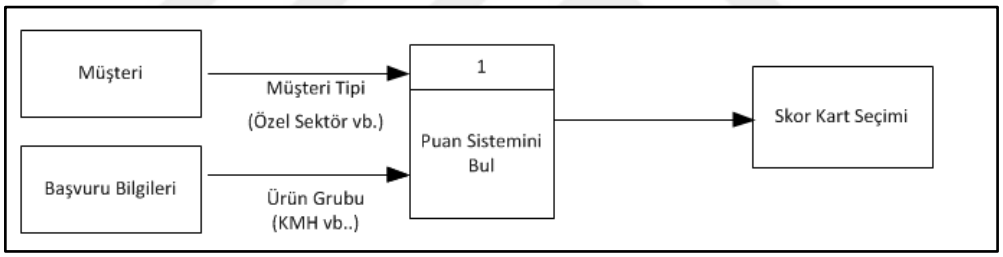
Veri akış diyagramı (DFD), sistem ve herhangi bir işlemin bilgi akışını görselleştirir. Veri akış diyagramları yeni bir modeli modellemek, sistemin genel bir görünümünü sunmak veya mevcut bir sistemi analiz etmek için kullanılmaktadır.

- Veri akış diyagramları sisteme giren bir verinin, sistemden nasıl çıktığını görmemizi kolaylaştırır.

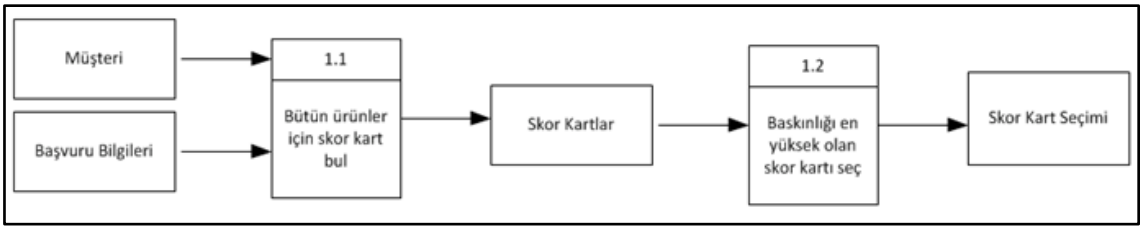
- Bu diyagramlar sayesinde birimler arası bilgi akışında nerelerde eksik, nerelerde gereksiz bilgi kullanımı var hemen görür ve müdahale etme şansını elde etmiş oluruz.
- Veri akış diyagramları analiz yapmamızı ve sistemi tasarlamamızı kolaylaştırır.
- Analizden tasarıma iyi bir köprü görevi görür.
- Kullanıcının sistemdeki her aşamayı kolayca görmesini ve anlamasını sağlar.
- Sistemimizdeki mevcut durumu görmemizi sağlar.



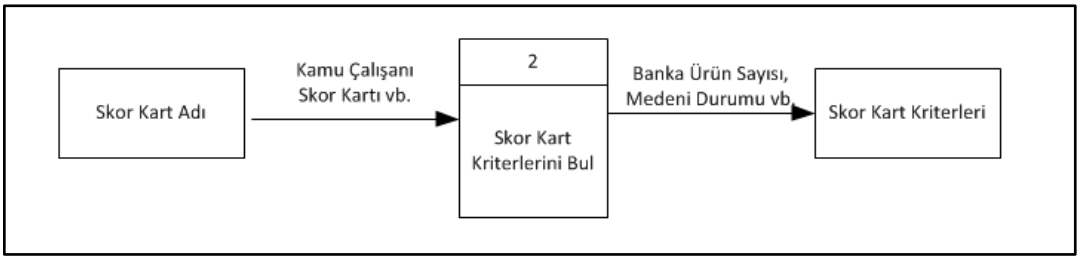
Şekil 3. 16 Veri Akışı Diagramı 0.Seviye



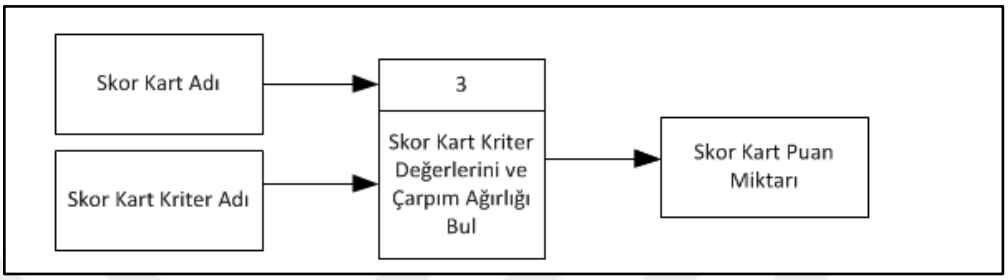
Şekil 3. 17 Veri Akışı Diagramı 1.Seviye



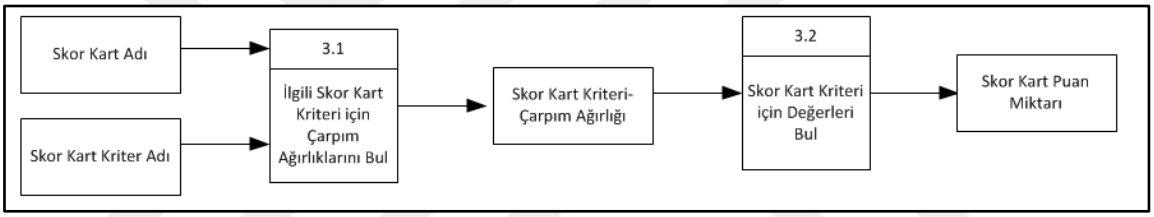
Şekil 3. 18 Veri Akışı Diagramı 1.1. Alt Seviye



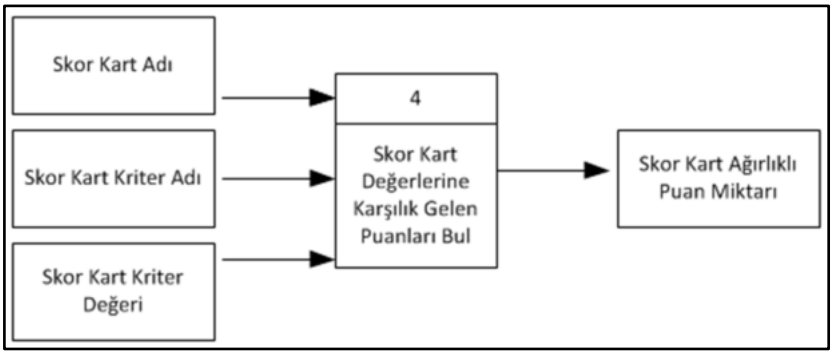
Şekil 3. 19 Veri Akışı Diagramı 2.Seviye



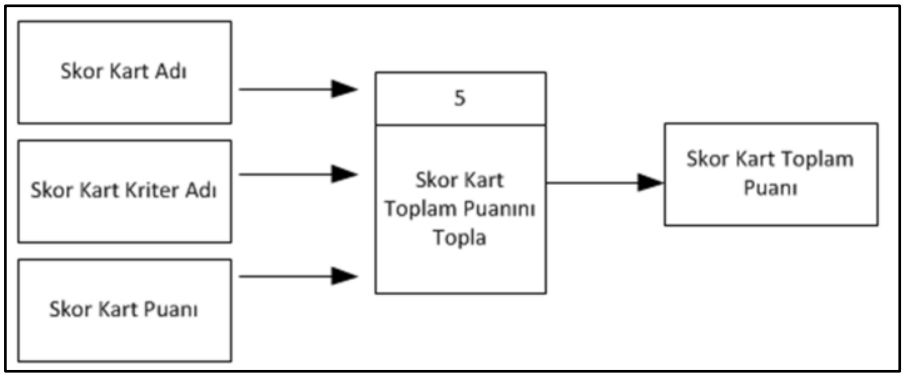
Şekil 3. 20 Veri Akışı Diagramı 3.Seviye



Şekil 3. 21 Veri Akışı Diagramı 3.1 Alt Seviye

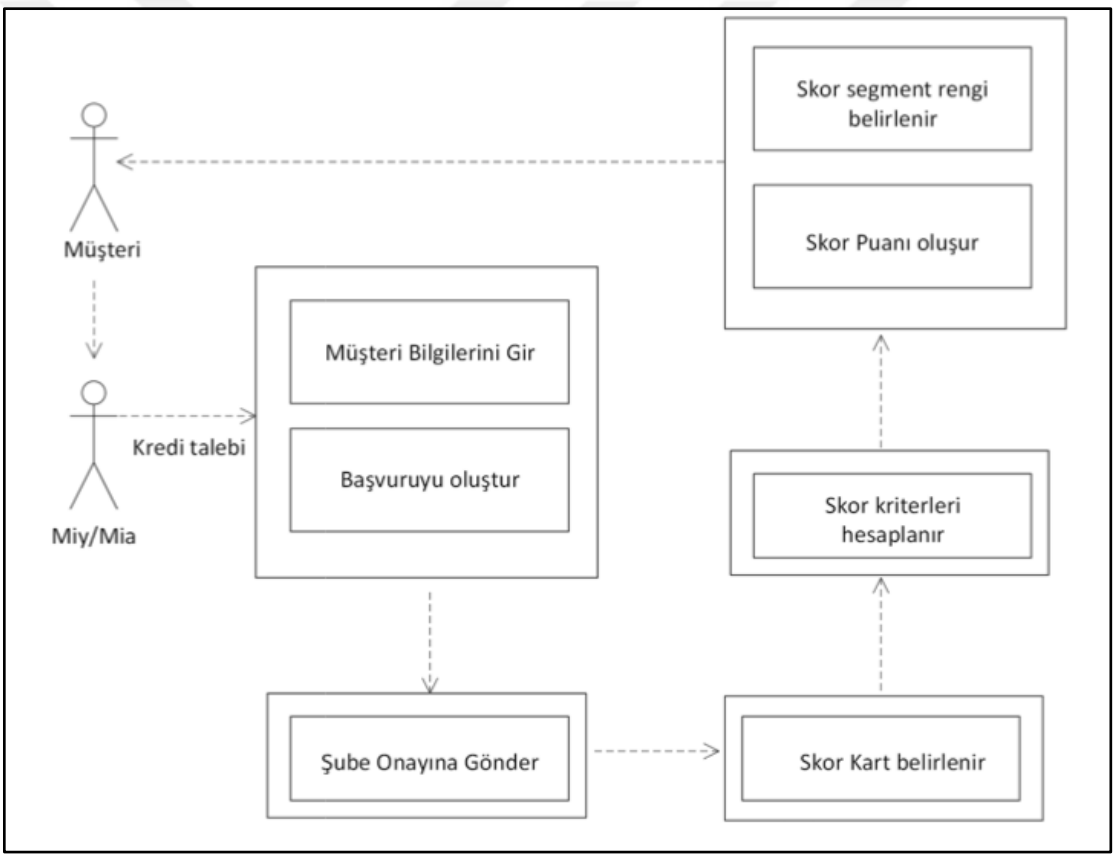


Şekil 3. 22 Veri Akışı Diagramı 4. Seviye



Şekil 3. 23 Veri Akışı Diagramı 5. Seviye

Sistemi modellemek için kullanılan data flow diagram bilginin nerelerden geçtiğini ifade etmek için kullanılmaktadır.



Şekil 3. 24 Aktör ve Sistem Arasındaki Bilgi Akışı

3.2.3 Sequence (Ardışık - Dizge)Diagram

Nesne etkileşim şemaları UML Modeli içinde Davranış Şemalarının bir parçasıdır. Genellikle Birlikteliklerin (Collaborations) incelenmesinde kullanılırlar. Nesne Etkileşim Şemaları nesnelerin birbirleriyle olan ilişkilerini bu nesnelerin arasındaki mesajlar ve nesnelerin birbirleriyle oluşturdukları işbirliği açısından ifade etmeye yararlar.

Sistemin çalışmasındaki akışı ifade eder. Çalışan objeler ve ilişkileri (iletişimi)ortaya koymak için kullanılır. Classların çalışma sırasında hangi durumlarda birbirinin metotlarını çağırdığını gösteren diagramlardır. Bir classtan birden fazla obje olabilir. Genelde bir süreç anlatılır her süreç ayrı çizilir. Akış içerisinde sıra ile gösterilir.

Ardışık sıra diyagramları ile nesnelerin birbirleri ile haberleşmesi zamana bağlı olarak ele alınır. Bir diyagramın zamana bağlı olmasından kasıt, nesnelerin gerçekleştirdikleri aktivitelerin peşi sıra gerçekleşmesi ve bu peşi sıralığın belirlenen zaman dilimleri içerisinde meydana gelmesidir. Ardışık sıra diyagramlarında yukarıdan aşağıya doğru inen hatlar bizim zaman çizgimizi gösteriyor. Soldan sağa doğru sistemde ilerleyen oklar mesaj ya da emirleri, sağdan sola doğru gelen oklar ise yanıtları belirtiyor.

Dizge diyagramları, tasarımcıya ya da ekibin bakış açısına, sistem gereksinimlerine ve modülün tüm paydaşlar tarafından bilinirliği başta olmak üzere birçok etmene bağlı olarak farklılık gösterebilir.

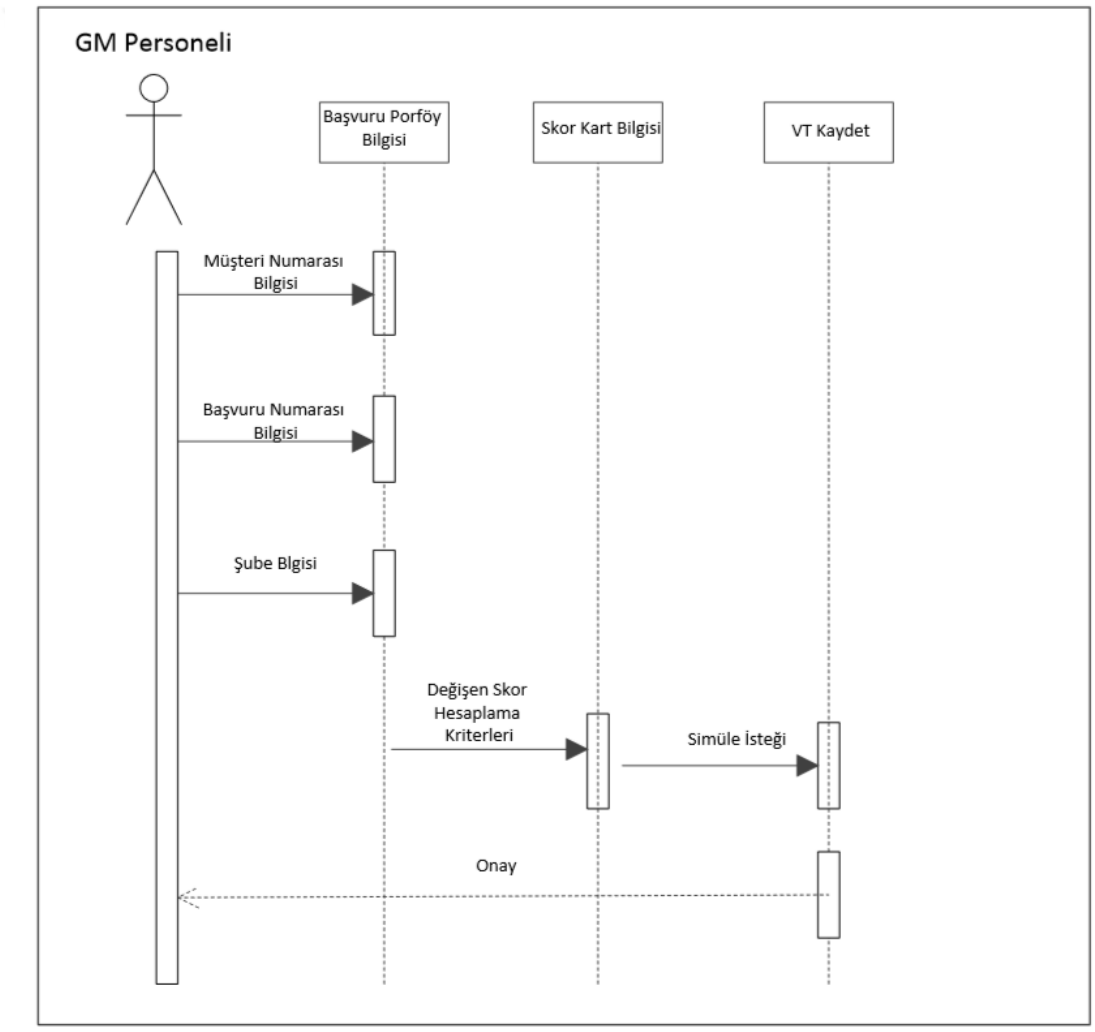
Kolayca yorumlanabilen sistemler için basit bir dizge diyagramı yeterli olabilirken, yeni geliştirilmekte olan bir sistem için her bir detayı gösteren diyagramlar gerekebilmektedir.

Bu ayrım gözetildiğine ardışık diyagramları iki türe ayrılabilir.

- Genellikle basit bir şemanın çizildiği ve en iyi olasılıkları ele alan “best case” senaryo üzerinden yürüyen örnek (instance) dizge diyagramı.
- Yazılım modelinin her yönüyle ele alındığı ve daha kompleks bir model olan genel (generic) dizge diyagramı.

Dizge diyagramları okunurken soldan sağa ve üstten alta doğru okunmaktadır. Dizge diyagramları kullanım durumlarıyla (use case) doğrudan ilişkili olup başlangıç noktasını ve aktörü belirlemektedir.

Her bir nesnenin altından çıkan kesikli çizgi zaman çizgisini ifade etmektedir. Kesikli çizgi zaman akışını belirtirken, üzerinde bulunan ince uzun dikdörtgenler o nesnenin zaman içerisinde meydana getirdiği eylemi temsil eder. Teorik olarak dikdörtgenin uzunluğu eylemin uzunluğu ile orantılıdır. Müşteri geliyor; başvuru yapılıyor; veri tabanına işleniyor; skor puanı yüksek olup olmamasına göre başvuru onay ya da red ediliyor.



Şekil 3. 25 Skor Kart Simülasyonu Sisteminin Dizge Diagramı

3.2.4 Class Diyagram

Sınıf diyagramları sistem gereksinimlerinin müşteriye anlatıldığı diyagramlardır. Temel olarak tasarımda kullanılır. Müşteri- çözümleyici iletişimini desteklemesinin yanı sıra tasarımcı ve programcıya da hitap etmektedir. Tasarımı ilgilendiren kısmı oldukça kapsamlıdır. Sınıf diyagramları aynı zamanda kendi başlarına oldukça sade yapıda olup, tasarım çizimleri kendi aralarındaki ilişkilerle anlamlı bir bütün oluşturmaktalar. Sınıf diyagramları sistemin statik yapısını tanımlamak için kullanılır.

Tasarım veya alan deseni kullanımında durum senaryoları oluşturduktan sonra bu senaryolar üzerindeki sınıfların ortaya koyulması gerekmektedir. Bu senaryolara dayanarak ve genelleme yaparak; müşteri, başvuru, kriter.. gibi sınıf seçilebilir.

Sınıf diyagramımızı oluştururken ilk önce sınıflarımızın özelliklerini, işlevlerini ve diğer sınıflar ile ilişkilerin belirlenmesi gerekmektedir. Örnek olarak Müşteri sınıfına bakmak gerekirse;

- Müşteri sınıfımızın özellikleri müşteri numarası, müşteri tipi, apartman, no, posta kodu, şehir ve ülkedir.
- Müşteri sınıfımızın işlevi ise onayladır.
- Müşteri – Yönetici arasında bire bir veya daha fazla ilişki var.
- Müşteri – Öğretmen arasında bire bir veya daha fazla ilişki var.
- Müşteri – Öğrenci arasında bire bir veya daha fazla ilişki var.

Sınıflar arasındaki ilişkiler aşağıdaki gibi listelenebilir:

1. Bire-bir ilişki

2. Bire-çok ilişki

3. Bire-bir veya daha fazla ilişki

4. Bire-sıfır veya bir ilişki

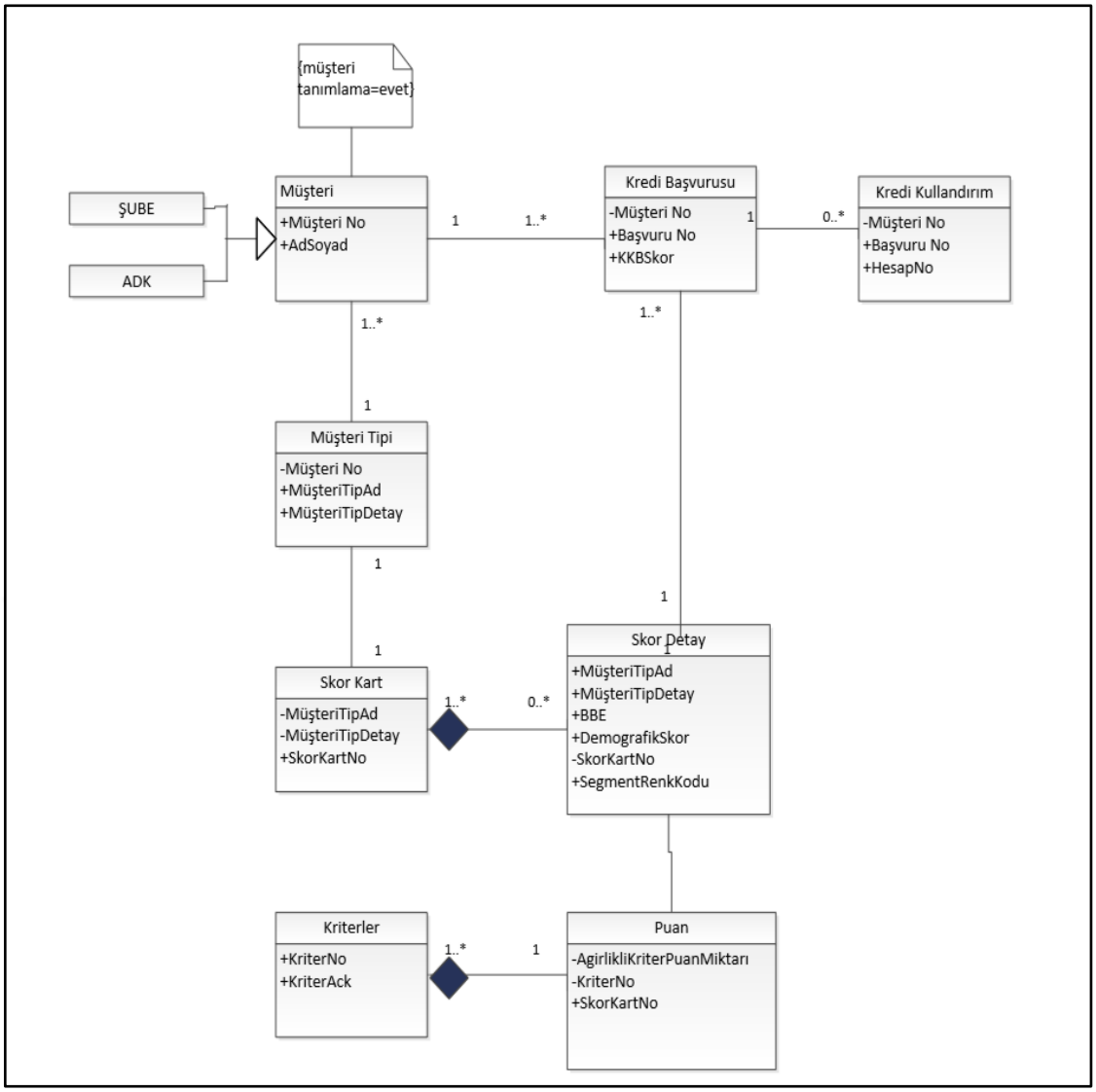
5. Bire-sınırlı aralık (mesela: bire-[0,20] Aralığı) ilişkisi

6. Bire-n (UML de birden çok ifadesini kullanmak için '*' simgesi kullanılır.) ilişki

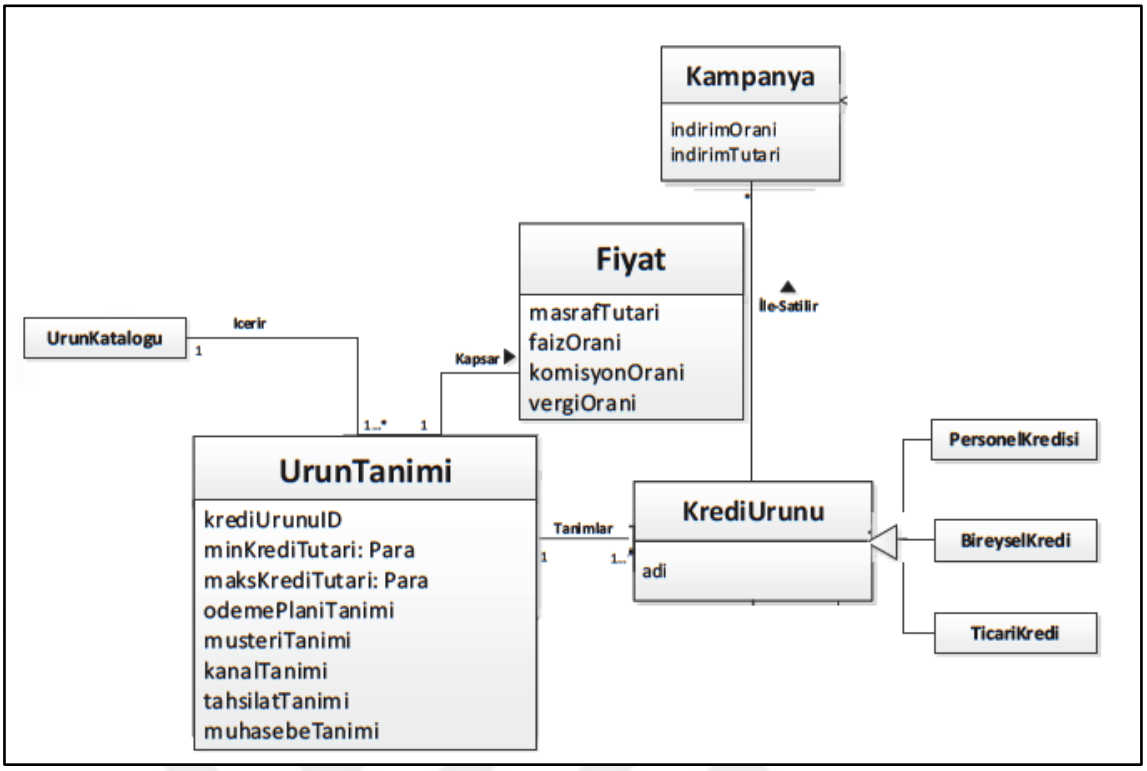
7. Bire-Beş ya da Bire-sekiz ilişki

Class nesne tabanlı yaklaşım içerisindeki en temel öğedir. Veri yapısı ve bu veriler üzerinde yapılabilecek işlemlerin benzeştiği durumlarda onların gruplanarak, bir soyutlama oluşturması işleminin sonucudur. Bu soyutlamadan (şablondan) üretilecek her nesne bu tanımlı değişken türlerinden başkalarını içeremez ve bu veriler üzerinde daha önce tanımlanmış işlemlerden başkalarının yapılmasına izin veremez. Class'ların içerebileceği iki temel bilgi türünden ilki değişkenlerdir (attributes, variables, fields). Değişkenlerin bir tipi ve program akışı içinde alabilecekleri değerleri vardır. Nesneler değişkenlerin çeşitli değerler taşıyarak, program akışı esnasında ihtiyaç duyulan veri alışverişini sağlarlar.

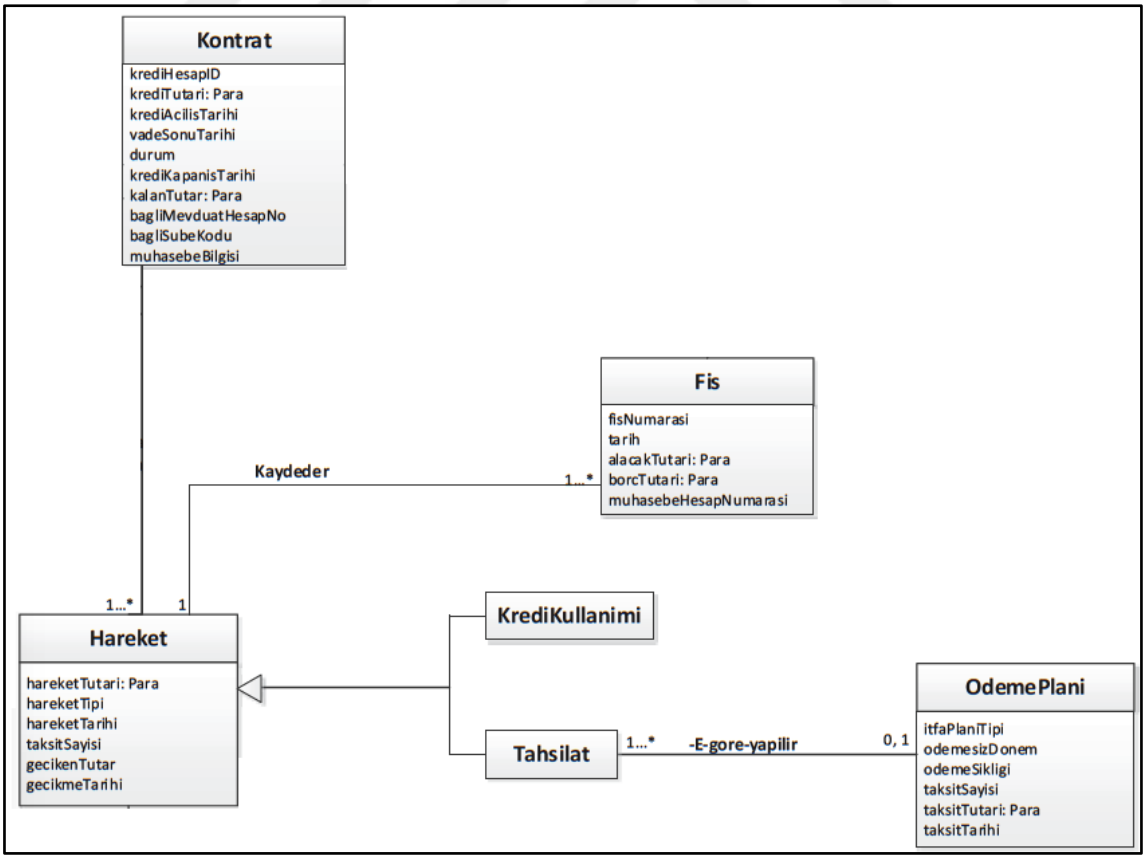
Değişkenler birer ilkel veri tipinde (integer, boolean, character, vs.) olabilecekleri gibi birer class tipinde de olabilirler. Dolayısıyla kendi tanımladığımız class'ları da birer değişken olarak program akışı içerisinde kullanabiliriz. Class'ların içerebileceği diğer bilgi tipi metotlardır (method, responsibility, function). Bu metotlar temel soyutlamaların tanımlandığı analiz aşamalarında birer sorumluluk olarak ortaya çıkıp, daha sonra tasarım çalışmaları esnasında detayları artık belirlenmiş (return type, parameter, precondition, postcondition, vs.) fonksiyonlara dönüşürler.



Şekil 3. 26 Kredi Skorlama sisteminde oluşturulan sınıfların birbiri ile ilişkisi



Şekil 3. 27 Ürün Açısından Sınıf Tanımı



Şekil 3. 28 Kredi Muhasebesi Açısından Sınıf Tanımı

3.2.5 Activity - Etkinlik Diyagramı

Activity Şeması zaman içerisinde birbirini takip eden faaliyetlerin ilişkilerini göstermek amacıyla kullanılır. Bu faaliyetler ardışık olabilirler, birbirlerini izlemeleri veya birbirlerine alternatif oluşturmaları belli koşullara bağlı olabilir. Bütün bu bilgileri belli bir detay seviyesinde ve tanımlı muhataplar tarafından tüketilmek üzere activity şemalarını kullanarak ifade edebiliriz.

Nesnelerin ve sistemin durumlarını etkileyen olaylar ve etkinlikler soyut bir düzeyde incelenmektedir. Nesnelerin programın kendi akışı içerisinde gerçekte nasıl bir işbirliği yaptığı, nasıl bir etkileşimde bulunduğu modellenmektedir.

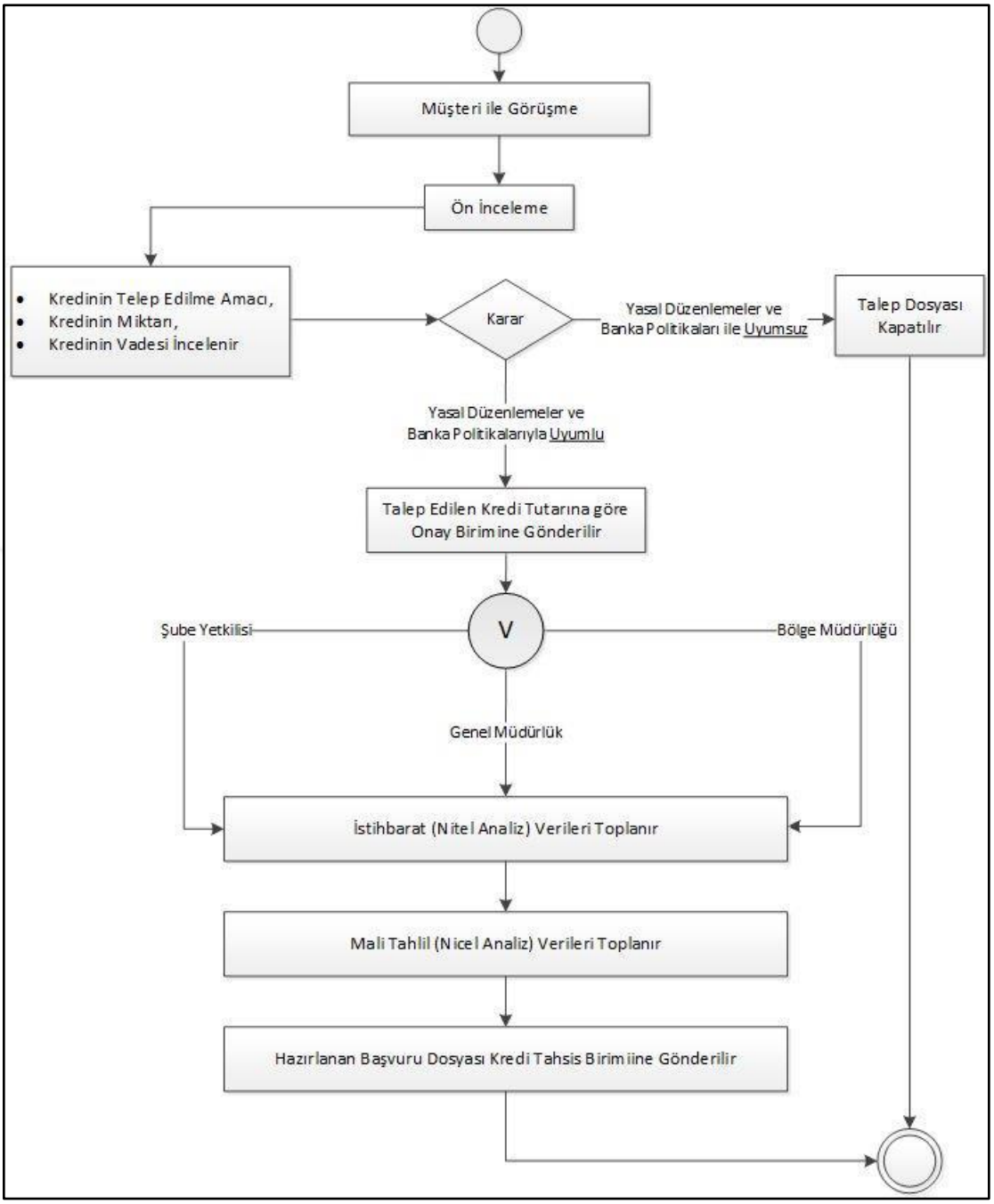
Activity şeması kullanım nedenleri arasında şunları sıralayabiliriz [65]:

- Use Case (Kullanım Senaryosu) akışlarını göstermek,

- Use Case seviyesinin de üstünde, tüm sistemi kapsayacak şekilde, bir "workflow" (İş Akışı) şeması yerine,

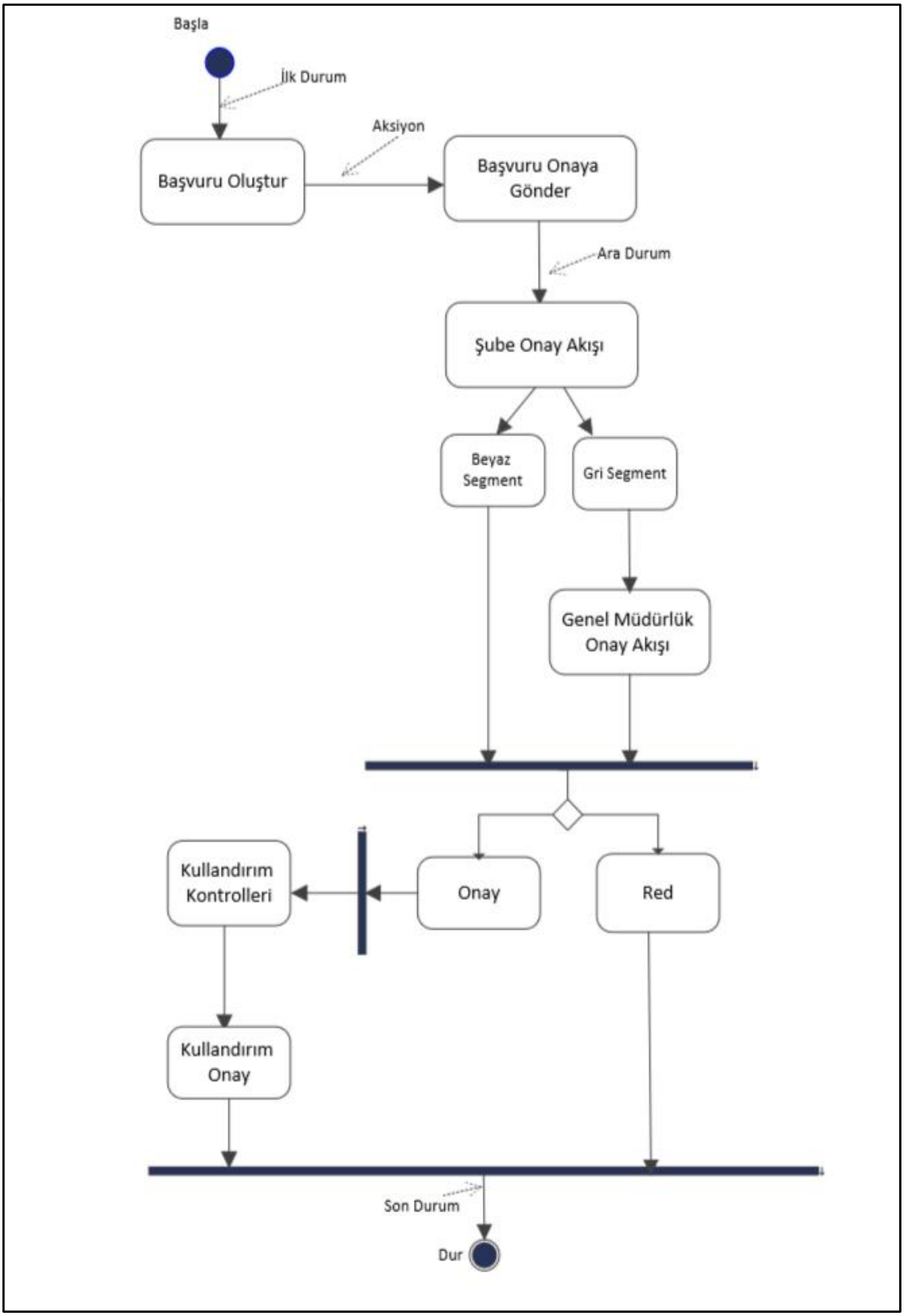
- Use Case Map oluşturarak uzun bir zaman diliminde tek başlarına gözlenebilir ve anlamlı faydaların (use case'lerin) nasıl bir ilişkiye sahip olduklarını göstermek amacıyla,

- Çok daha detay seviyede karmaşık bir algoritmanın hangi komutlar ve mantık dahilinde gerçekleştiğini göstermek amacıyla kullanabiliriz.



Şekil 3. 29 Kredi Başvuru Süreci [67]

Kredilendirme süreci, Şekil 3.29'da olduğu gibi müşteri talebi ile başlar [67]. Bu talep bazı durumlarda banka çalışanının müşteriyi ziyaretiyle de oluşabilir. Bankanın ilgilendiği ve yakın markaja aldığı marka sahipleri ya da ortaklarıyla ihtiyaç görüşmeleri yapılır ve finansal değerler masaya yatırılır. Bu araştırmada kredi başvurusu yapan şahsın geçmiş dönemlere ait performansı dışında geleceğe yönelik beklentilerinin de analizi yapılır.



Şekil 3. 30 Kredi Skor Puanı Oluşumu Etkinlik Diagramı

3.3 Ekran Prototipinin Belirlenmesi

Artık günümüzde tasarımcılar Adobe XD, Sketch, Invision gibi programlar ile kodlama bilmeden sanki ürün bitmiş gibi ürünü çizebilir ve akışları da oluşturarak kullanıcı deneyimini test edebilmektedir. Bu sayede daha olmayan ürünü bile hedef kitleye test ettirebilir ve nerelerde takıldığı görülebilmektedir. Bu araçlar sayesinde ürün yöneticisi de kullanıcı deneyimini daha yazılımcı koda başlamadan görmekte ve zaman kaybı önlenmektedir. Bu araçların bir faydası da yazılımcının işini kolaylaştırmasıdır. Ürünle ilgili isteklerinizi ne kadar yazarak anlatırsanız anlatın tasarımı bitmiş bir üründen geriye gelerek aktarmaktan daha kolay değildir. Tasarım ve prototip aşamasında tasarımcıya ne kadar çok done verirseniz ve ne kadar çok vizyonunuzu paylaşırsanız o kadar iyi bir ürün ortaya çıkar ve yazılımcı da bir o kadar kolay geliştirme yapar [60].

Skor kart yönetim ekran prototipinde iş mantığı gereği müşteri kredi başvuru talebinde bulunurken müşteri tipine bağlı skor kart tanımı üzerinden skor puanı hesaplanacak şekilde tasarlanacaktır. Müşteri tipi seçimi ile emekli olmasına bağlı emekli skor kartı; kamu çalışanı olmasına bağlı kamu çalışanı skor kartı, özel sektör çalışanı olmasına bağlı özel sektör skor kartı ve diğer müşteri tipine bağlı(serbest meslek, firma sahibi gibi)diğer skor kartı varsayımının kabulü ile belirlenecektir. Skor kartın belirlenmesi ile müşterinin demografik puanı, KKB puanı, BBE bilgisi gibi tüm kombinasyon seçimlerinin varlığı üzerinden kredi onay sürecini etkileyen segment rengi sınıfı oluşur. Her skor kartın puan değerinin hesaplanmasında esas alınan kriterler belirlenir.

Skorkart hesaplama kriterlerinden bazıları:

- Oturulan ev durumu(ev sahibi, kira, lojman)
- Talep edilen kredi türünün(taşıt,tüketici,konut) vadesi
- Bankamız vadeli mevduat tutarı
- Meslek (serbest meslek, çiftçi..)
- Yaş
- Aktif olan kredisine bağlı gecikme durumu
- Son 6 ay içerisinde kapanmış kredisine bağlı gecikme durumu
- KKB son 12 ay kapalı kredi takip sayısı
- KKB diğer banka gecikme durumu

- Medeni durum
- Son işyerinde çalıştığı süre

Her kriter, karar vericinin zaman içerisinde belirlediği değerler ölçütünde belli bir puana ve ağırlıklı çarpım miktarına karşılık geldiği varsayılmaktadır.

Skor Kart Puan Aralıklarını Etkileyen Segmentler

KKB Skoru: KKB nin esas aldığı kriterler üzerinden iletilen puandır

Demografik Puan: Bankanın belirlemiş olduğu onay durum segment rengine göre karar verilen puan ölçütü

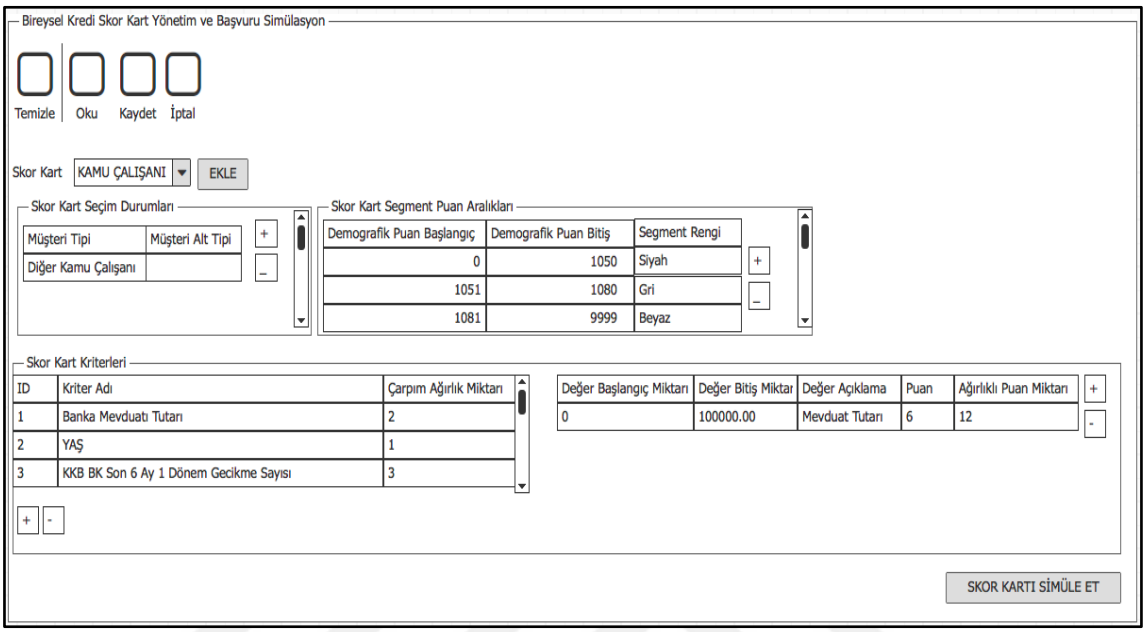
BBE: Skor bazlı, ileriye dönük aşırı borçlanma ihtimalini gösteren bir risk endeksi olarak tanımlanmaktadır. Son dönemde ve geçmişinde ödeme güçlüğü belirtisi göstermeyen, ancak ödeme gücünü aşan bir borç yükü altına girme eğilimindeki kişileri öngörmeyi hedefler. Böylece, mevcut borçlarını geciktirmeden kapatmak için yeni krediler kullanan, ancak gittikçe daha fazla borçlanan bireyleri tespit ederek, kişinin borçlanma eğilimini tek bir endeks puanıyla ifade eder**.**

Karar verici açısından esas önemli konu kredi skorlama puanının oluşum kriterlerini yönetmek, yapılacak değişikliklerin gerçek bir veri havuzunda belli bir dönem baz alındığında etkisini sahada gerçekleştirmeden simülasyonunu yapabilmek ve raporlayabilmek olmaktadır. Simülasyon sonucunu analiz eden karar verici örneğimizde olduğu gibi yaptığı değişikliği gerçek sistemde uygulamaya alabilir ya da uygulamaya almaktan vazgeçebilir. Karar vericinin bu sisteme ihtiyaç duyması uygulamaya almak istediği değişikliğin sahaya etkisini devreye almadan göstermesini sağlamakta ve bankanın kredi verme onay mekanizmasını doğrudan etkilenmesine imkan vermektedir.

Kredi verme onay mekanizması çalışma kapsamında ele alınan uygulamamızda segment rengi bazlı ifade edilecektir.

Kredi skor puanı hesaplanmasında kredi değerlendirme kriterlerinin oluşturduğu puanların toplamı belirlenen aralıklar ile beyaz, gri ve siyah segment olarak ifade edilmektedir.

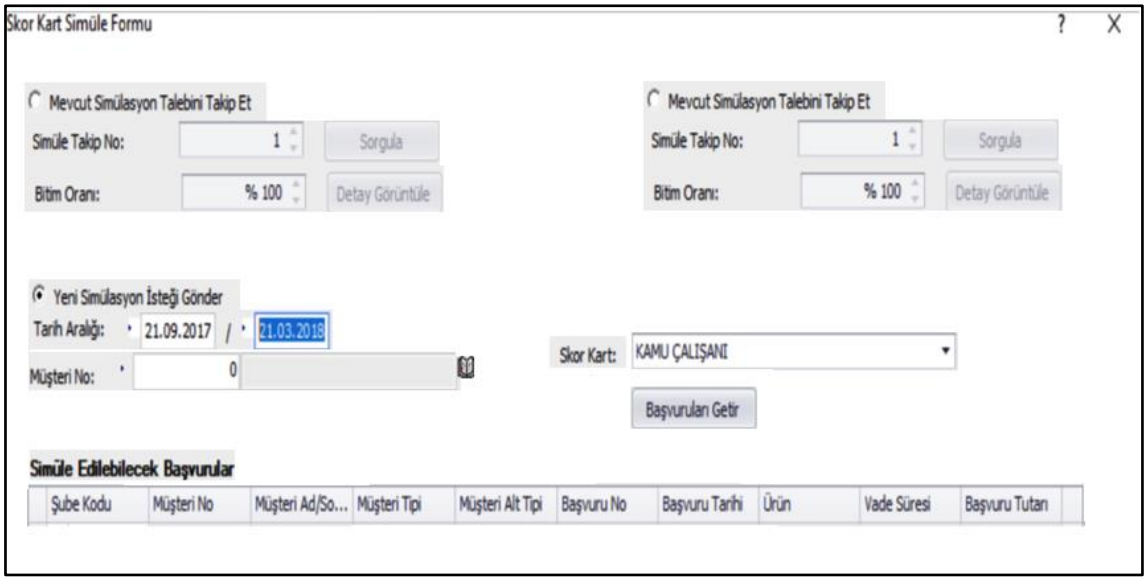
Sözgelimi beyaz segment rengine sahip olan bir müşteri kredibilitesi yüksek bir müşteri olduğunu göstermekte iken, gri segmente sahip bir müşteri üst bir tahsis değerlendiricisi tarafından değerlendirilmesi anlamına gelmekte, siyah segmente sahip müşteriler ise red edilen müşteri sınıfını ifade etmektedir.



Şekil 3. 31 Kamu Skor Kartı Yönetimi Ekran Prototipi

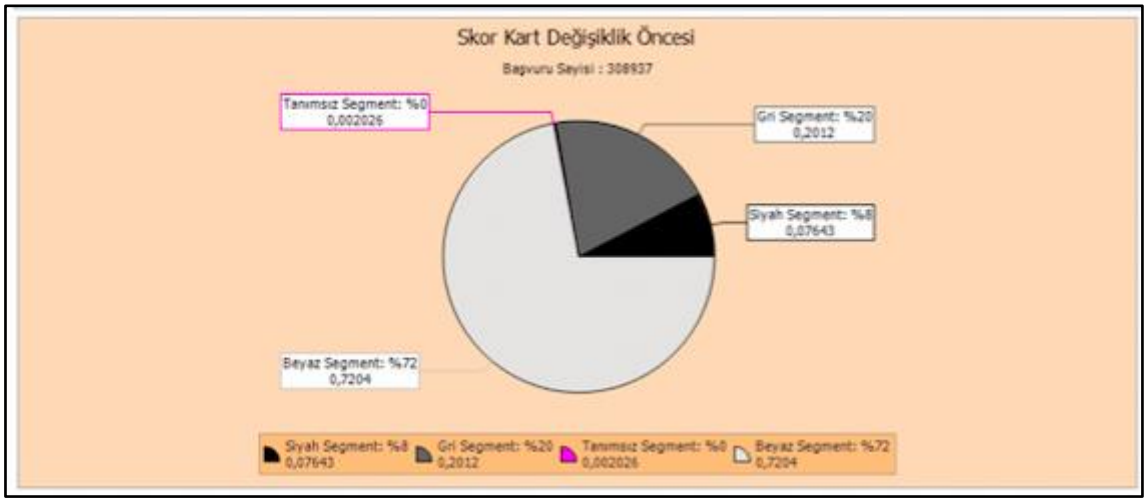
Tasarlanan ekran üzerinden aynı zamanda raporlama ihtiyacını karşılamak adına kredi skorlama siteminde yapılacak herhangi bir değişiklik için simülasyon isteği gönderilmekte ve çalışan bir batch yardımı ilgili müşteri grubuna/müşteriye değişikliğe neden olan durumun karar değerlendirmesi yapılmaktadır.

Tasarım deseni ve çevik yöntem kullanılmadan önce bu durum simüle mantığı olmadan işletilmekte ve karar vericinin yanlış karar vermesine neden olmaktaydı. Aşağıda prototip form üzerinden simüle isteğinin nasıl oluşturulduğu ve takip durumu gösterilmiştir.

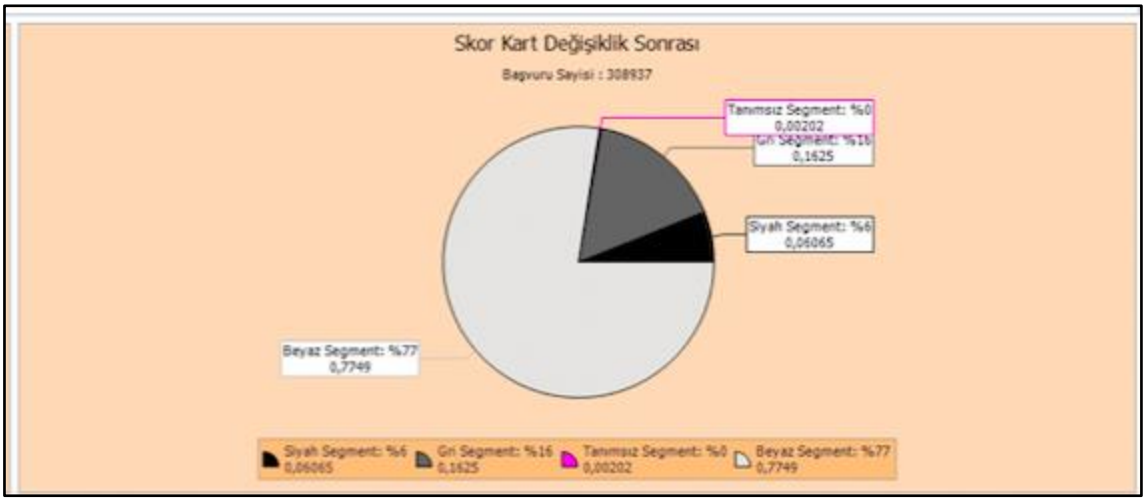


Şekil 3. 32 Kamu Skor Kartında Simüle İsteği Formu Prototipi

Örnek bir kamu çalışanı müşteri tipine bağlı skor kart yapısında iyileştirme yapmak isteyen kullanıcı, vadeli mevduat tutarına bağlı olarak puan üzerinde artı yönde değişiklik yaptığında bu etkinin sahada ne ölçüde cevap vereceğini gözlemlemek için simüle isteği göndermektedir. Simüle isteğinin bitim oranı %100 e ulaştığında ise raporlama sonucu aşağıdaki gibi karşılaştırılmakta ve müşterilerin beyaz segmette düşme oranında artış görülmektedir.



Şekil 3. 33 Kamu çalışanı skor kartında yapılan iyileştirme öncesi



Şekil 3. 34 Kamu çalışanı skor kartında yapılan iyileştirme sonrası

3.4 Performans Analizi

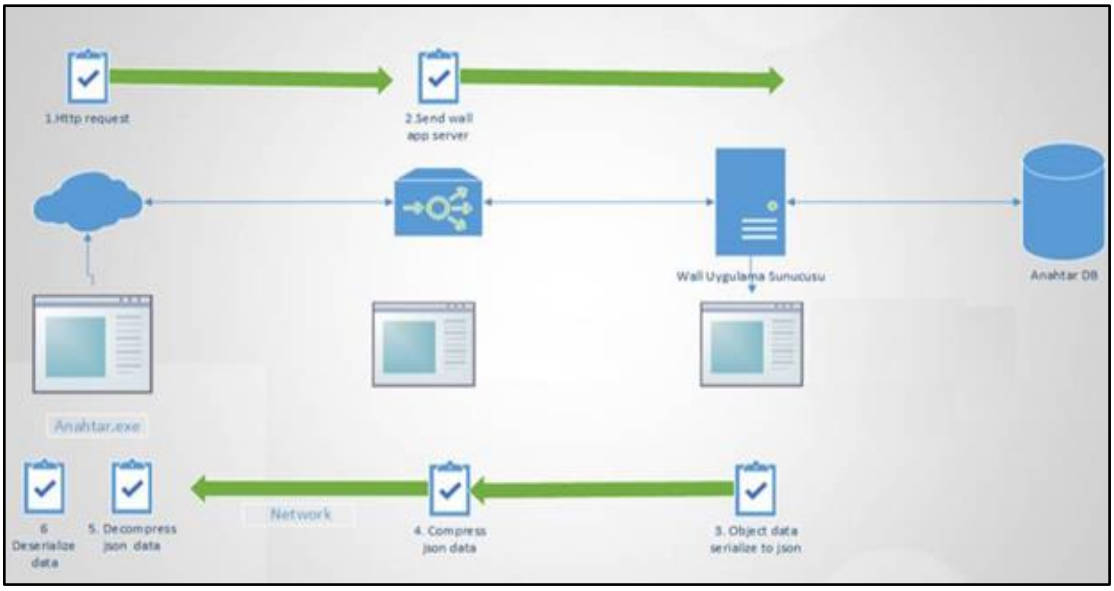
Waterfall akış ile yazılan kodun performans değerlendirmesinin çevik akış ile geliştirilen ve tasarım modeli kullanılarak yazılan koda göre performansı IIS(Internet Information Services) logları incelenerek karşılaştırılacaktır. Server isteklerinin saatlere gruplandığında ne kadar sürede cevap verdiği, kodun çalışmasında geçen sürenin ne kadar değiştiği, prosedür çağrıma isteklerinin ne olduğu ve ne kadar sürede cevap verdiği gibi sorulara cevap arayarak çevik yöntemi ile yazılan kodun performans değerlendirmesi yapılacaktır. Kullanılan uygulamalar uygulama sunucusundan çekilmek istenen istek sayıları ile sisteme gelen yükle birlikte değerlendirme imkanı verecektir.

Performans ölçümü yapılabilmesinin en büyük avantajı ise yeni yazılan kodun Restful Web Servise ile yazılmış olmasıdır. Rest HTTP'i protokolü ile yazılmış client-server iletişimi ile ilgili bir mimari olarak bilinmektedir. Rest mimarisi http isteğinde(request'inde) yapılması istenilen işlemin http metotları (POST, PUT, DELETE, GET) ile ifade edilmesi proxy ihtiyacını ortadan kaldırmakta ve bu sayede platformdan bağımsız(Client'ın Windows, Server'ın Linux olması gibi) yapı kurulmasına olanak sağlamaktadır. Restfull servisinde farklı cevap (response) tipi; (Json,xml,CSV, vb)olabilir.

Geçirilen süreç içerisinde eski yazılan kodun restfull mimarisi ile yazılmamış olması client'ın server tarafındaki veri kaynağı hakkında hiç birşey bilmemesine,Server'ın da doğru istekler için doğru yanıt verme durumu hakkında belirsizliğe neden olmaktadır. Yeni yazılan kodun web tabanlı olması sayesinde bu inceleme yapılarak kod kalitesi analiz edilebilecektir.

Bir başka açıdan performans kıstası olarak Http response'ları client tarafında cache'lenebilir olması, server'a giden yükün azalmasına neticesinde bu durum da network sıkışmasını engellemiş olmaktadır. Performans kriterlerinde diğer bir önemli konu ise Idempotent kavramıdır. Matematik veya Bilgisayar biliminde bir fonksiyonun bir defa uygulanması ile birden fazla uygulanması sonucu değiştirmiyor ise bu fonksiyon Idempotent olarak adlandırılmaktadır. Http metotları içerisinde yapılan incelemelerde kullanım açısından POST metotlar hariç tutulmaktadır. İncelemelerimizde apılan HTTP request'i için çağrılan URL ile beraber HTTP metotlardan biri seçilir ve sunucu yapılan talebin kayıt üzerine nasıl etki edeceğini buna göre belirlemektedir.

REST ile yazılmış bir servisle çalışmak için ihtiyacınız olan tek şey URL(web sunucusundan talep edilen sayfa / servis adresi). Bir URL'yi çağırırsınız, URL size JSON veya XML döndürür, dönen cevabı parse edersiniz ve servis entegrasyonunuz tamamlanır. Yani teorik olarak istemci uygulama REST bir servisin yapısını ve detaylarını bilmek zorunda değildir. REST'in bu basit standartları dışında uyulması gereken bir kural yoktur, son derece esnek bir yapısı vardır.

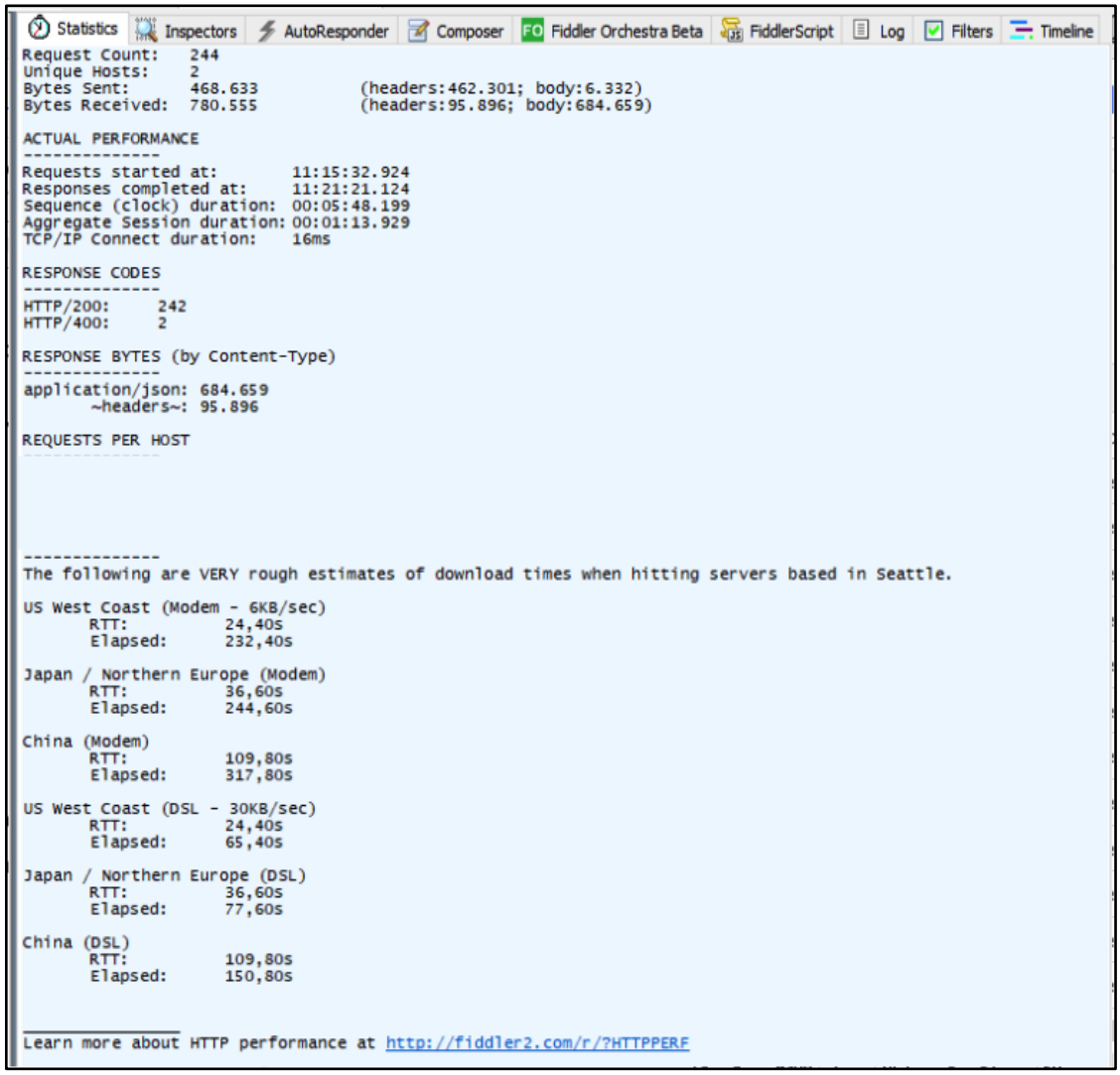


Şekil 3. 35 Anahtar Wall Request Flow

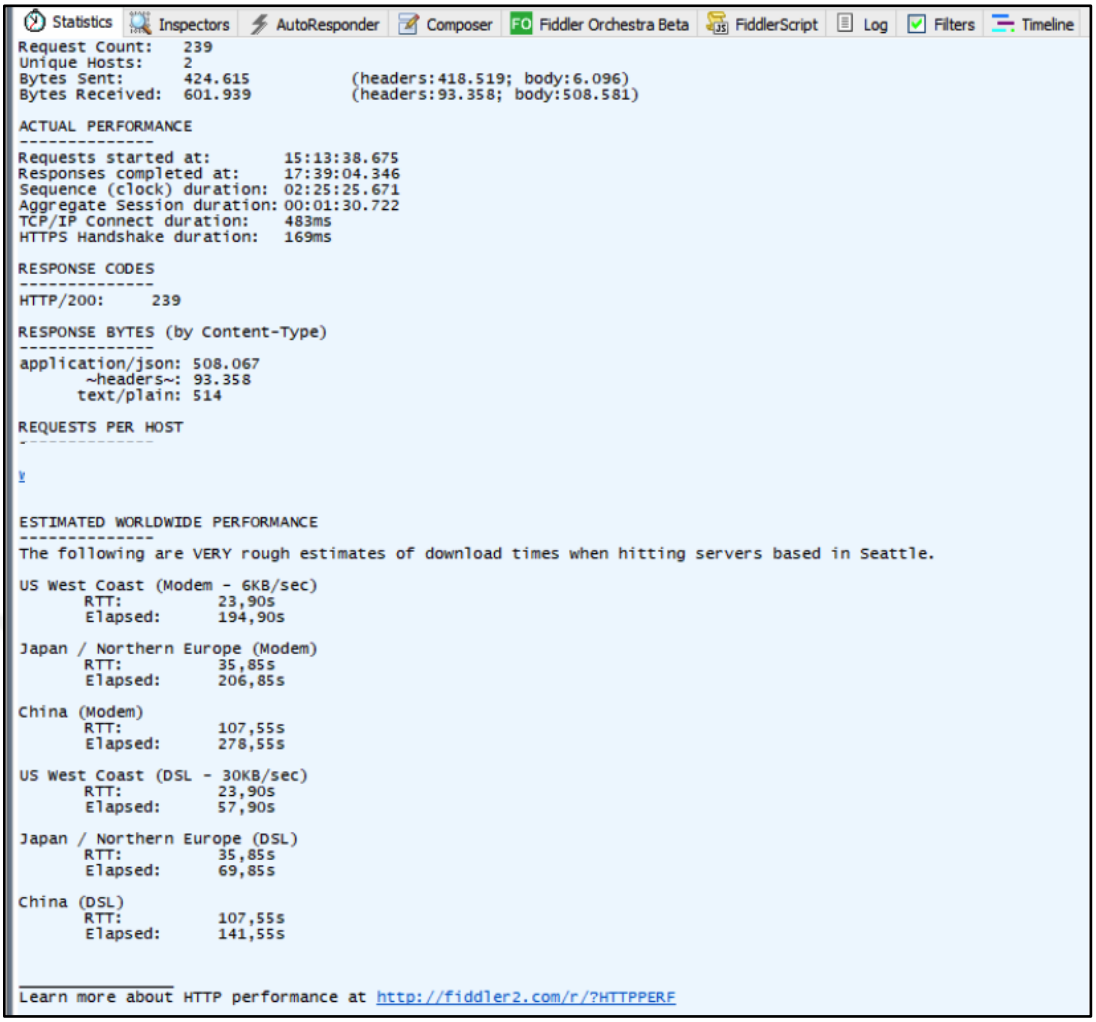
3.4.1 Fiddler İle Performans Analizi

Fiddler, bilgisayar ve internet arasındaki tüm web trafiğini takip eden ve detaylı bir şekilde analiz yapmamızı sağlayan ücretsiz bir http 'Hata Ayıklama' programıdır. Bu program, http trafiğini incelememiz, hangi işlemin ne kadar zamanda sürdüğü,request sayısı, veri çekme boyutu, hata alan işlem adımları gibi istatistiksel verileri de izlememize imkan vermektedir. Http session listesi ve işlem detayları görünen loglar analiz edilerek ekran performans ölçümü yapılmaktadır. işlemlerimize ait istatistikler(bağlantı zamanı, ne kadar zamanda yüklenme süresi oluşmakta gibi), requestleri düzenleme, trafik yoğunluğunu gözleme bu sayede analiz edilebilmektedir. Bununla beraber Breakpoint mantığı ile http/https trafiğini durdurup, işlemleri adım adım takip edebilir veya edit edebilme imkanı sunmaktadır. Breakpoint sayesinde Web site üzerinde istek gönderiminde database verilerini getirmeden durdurmak mümkün olmaktadır.

Şekil 3.36 ve 3.37 da görüleceği üzere yapılan parametre cache'leme mantığı sayesinde en büyük fark network'e gönderilen veri boyutu olmaktadır. Verinin boyutunun bu iyileştirme ile küçülmesi network'te geçen süreyi de azaltacak ve network'ün daha az satüre olması yani belli bir doyum noktasına ulaşması sağlanmış olacaktır.



Şekil 3. 36 Performans İyileştirme Yapılmadan Önce Fiddler İstatistiksel Sonucu



Şekil 3. 37 Performans İyileştirme Yapıldıktan Sonra Fiddler İstatistiksel Sonucu

Uygulamamız kapsamında çevik yöntem ile waterfall yöntem kullanılarak yazılan kodun ekran üzerinden istek gönderiminde çalışan url bazında istek çağrımının ne ölçüde olduğu konusunda analiz çalışması yapılmıştır. Amaç waterfall ile yazılan kodun incelenerek birden fazla istek çağrımına neden olan süreci, çevik yöntemde tekrarlı istek çağrımı yapmadan tek seferde daha performanslı bir istek çağrımının nasıl oluşturulabileceğini gözlemlemek olmaktır.

Ayrıca Application Insights ile da remote(varsa dış kurum bağlantısında geçen süre-kkb gibi), database de geçen süre, application code(yazılan kodun performansında geçen süre) gibi server da yada network tarafında response süreleri karşılaştırılmıştır.

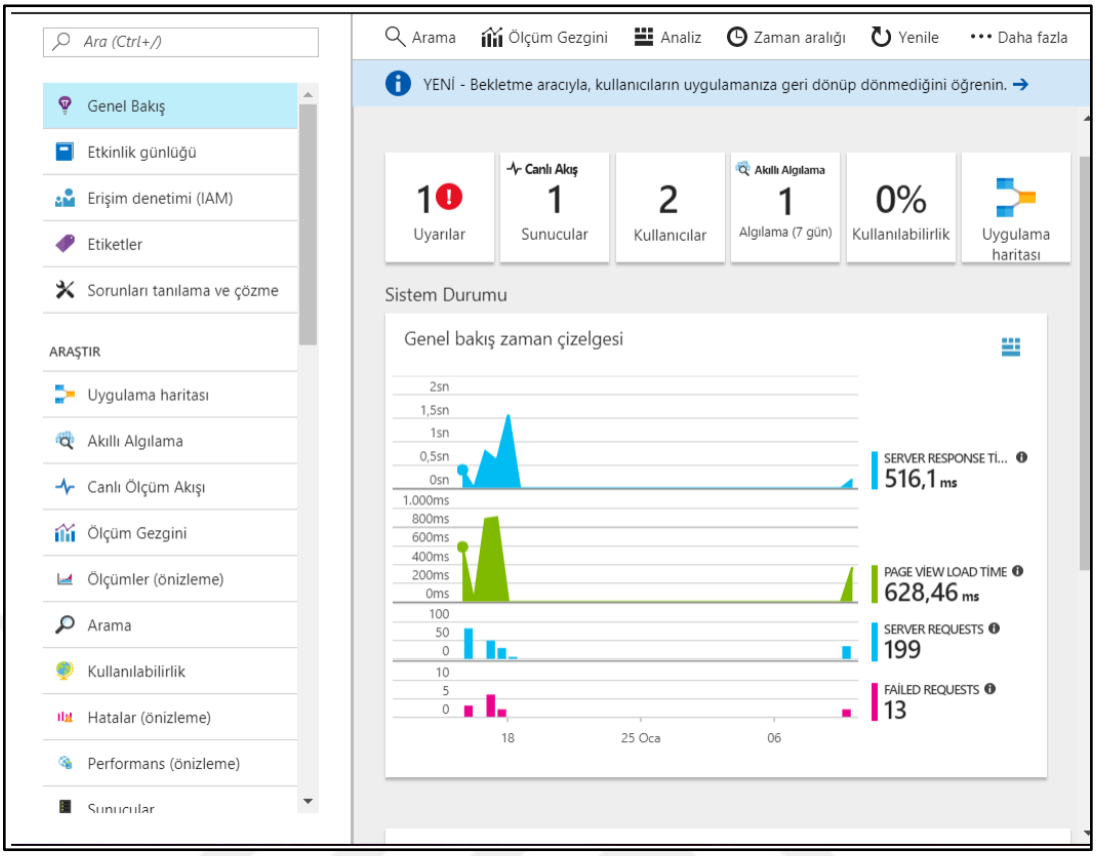
Burada amacımız yeni yazılan sınıfın performans değerlendirme analizini yapabilmek, client tarafında kullanılan uygulamanın ekran açılış command(oku-kaydet vb)sürelerini ve gönderilen isteklerin incelemelerini yapabilmektedir.

3.4.2 Application Insights İle Performans Analizi

Azure tarafından geliştirilen monitoring tooludur. Yapıda karşılaşılan hatalar ve performans sorunlarını anlık ya da geçmişe dönük detaylarıyla inceleyebiliyoruz. Yapının kullandığı kaynak durumu, varsa yapıda karşılaşılan darboğazların tespit edilmesi ve gerektiğinde müdahale edilmesi konularında hem zaman kazandırıyor hem de yapının daha kaliteli bir hal almasına katkıda bulunuyor.

Application Insights, çeşitli platformlarda çalışan web geliştiricilerine sunulan bir Uygulama Performans Yönetimi(APM) hizmetidir. Uygulamanızda meydana gelen sorunları tanımlamanıza, gelen telemetrileri yorumlamanıza, bağlantı sürelerini kontrol etmenize imkan sağlayan güçlü analiz araçları içeren bir hizmettir.

Uygulamanıza en büyük katkısı performansınızı ve kullanılabilirliğinizi sürekli geliştirmenize olanak sağlamasıdır.



Şekil 3. 38 Application Insights' ın Arayüzü

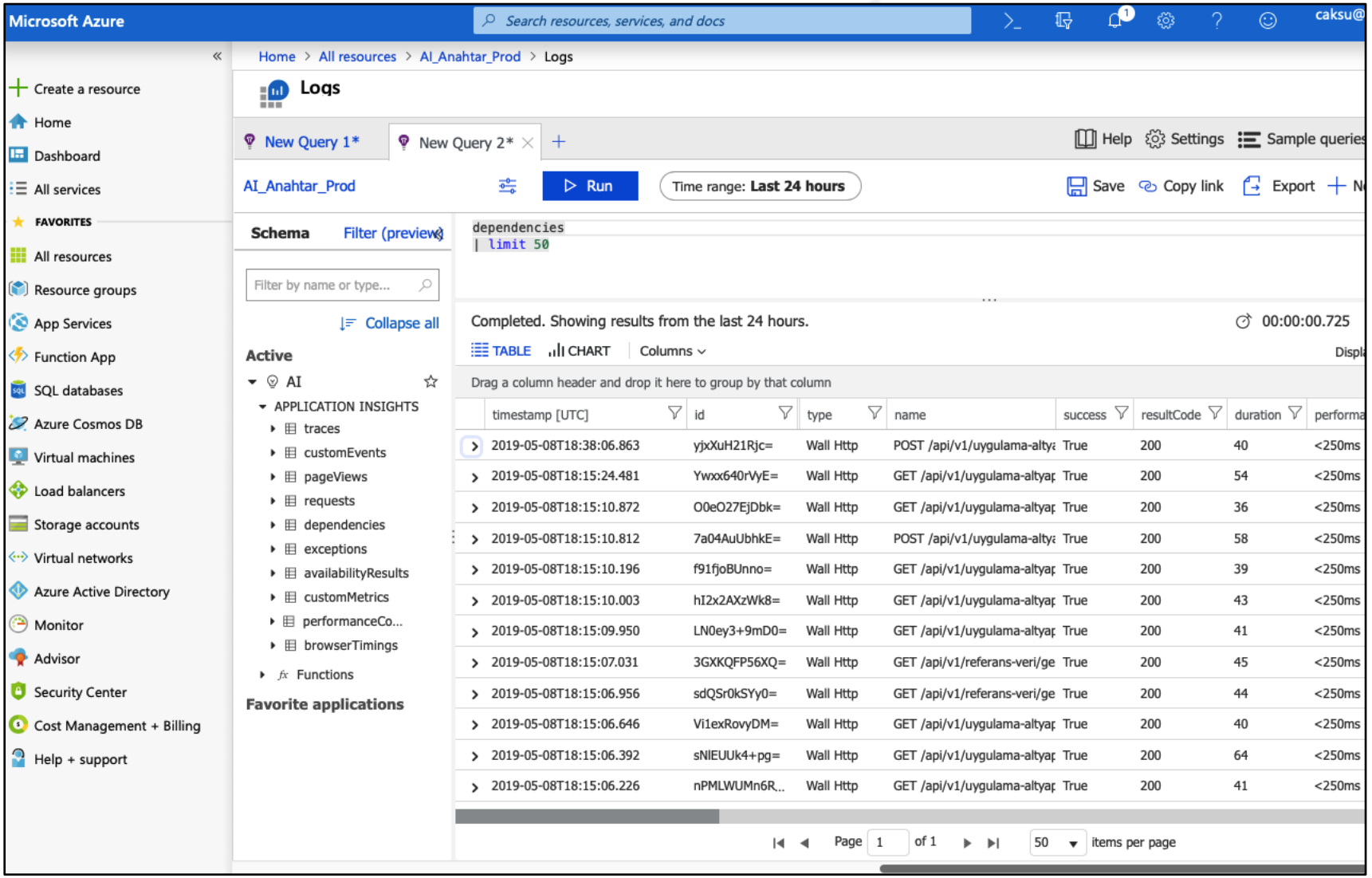
Application Insight'ın sağladığı fayda:

- Uygulamaya ve işlemciye ait bilgi toplanmasını görselleştirerek anlaşılır bir şekilde ortaya koyar,
- İş yüklerine ait sorunları önceden tespit ederek sahaya alınmadan önce alınacak riski azaltır,
- Uygulamamızın alt yapı kalitesinin izlenmesi ve yönetilmesine olanak sağlar,
- Sunucu güvenliği ve durumu hakkında bilgi vererek veri toplanmasına olanak verir,
- Hatanın kök nedenin tespit edilmesini sağlar,
- Son kullanıcı gereksinimlerine ne kadar cevap verdiğini ve kullanıcıya etkisini ortaya koyar.

Application Insights Veri Toplama ve Saklama Özelliği:

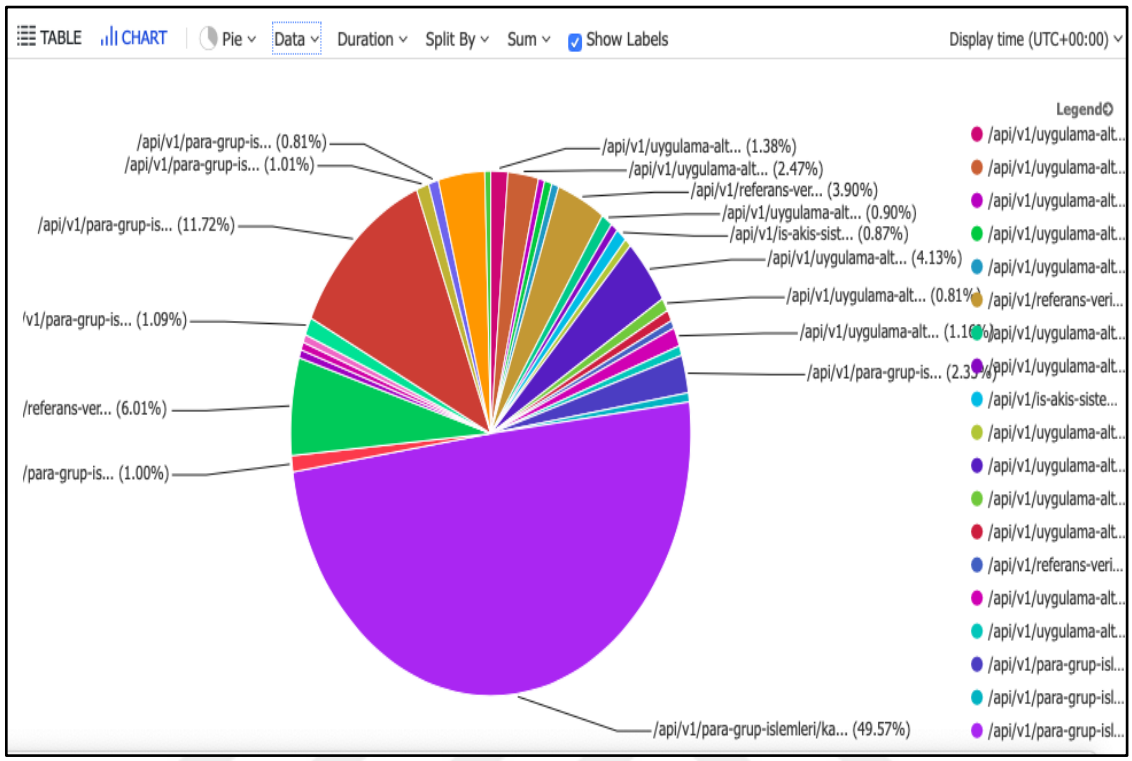
- Application Insights .Net / .Net Core / Java / Node.Js / Mobile ortamlarında kullanılabilir. Toplanan verilerin özelliği:
- Web server telemetry — HTTP istekleri, URİ, İstekleri işlerken geçen süreler, Yanıt kodları, Client IP adresleri, Session id
- Web sayfaları — Sayfa, kullanıcı ve oturum sayfaları, Sayfa yükleme süreleri, İstisnalar, Ajax calls
- Performans Sayaçları — Bellek, İşlemci, IO, Ağ doluluk değerleri
- İstemci ve Sunucu — OS, Yerel, Aygıt türü, Tarayıcı, Ekran çözünürlüğü
- İstisnalar ve Crashler
- Bağımlılıklar — REST, SQL, AJAX, URİ ya da connection string, süre, başarı,komut
- Kullanabilirlik testleri — Test süresi ve aşamaları, yanıtları
- Günlükler ve özelleştirilmiş telemetriler

Bu uygulama sayesinde command (OKU, ÇALIŞTIR, KAYDET vb) içerisinde yapılan çağrımlar, ekran açılış süreleri, http istekleri (dependencies), url bazında çağrım sayısı gibi incelemeler üzerinden sistem üzerinden performans iyileştirmeleri yapılmıştır. Bu amaçla her işlem için unique bir parent id oluşturulmuş ve müşteri için önemli bilgiler ile query'ler hash'lenerek gruplama yapılmıştır.

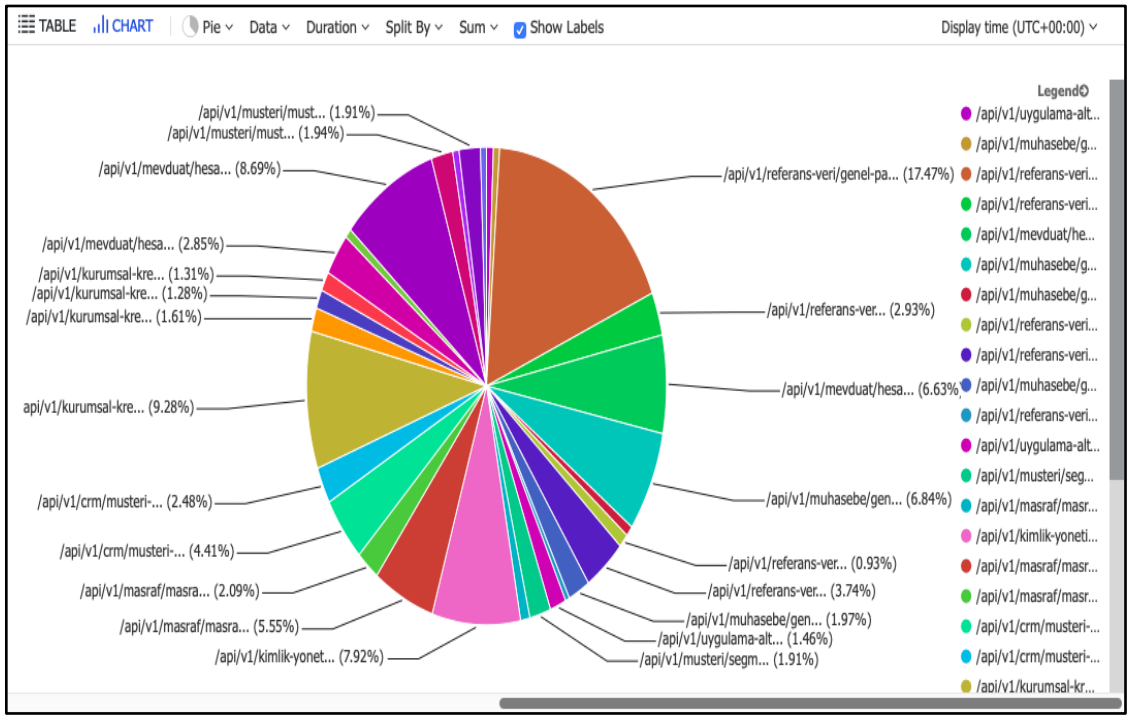


Şekil 3. 39 Azure Log Kayıt inceleme

Şekil 3.39 ve 3.40'da görüleceği üzere yeni yazılan kodun ilgili url çağrımı için yüzdesel oranda iyileşme analizi elde edilmiştir.



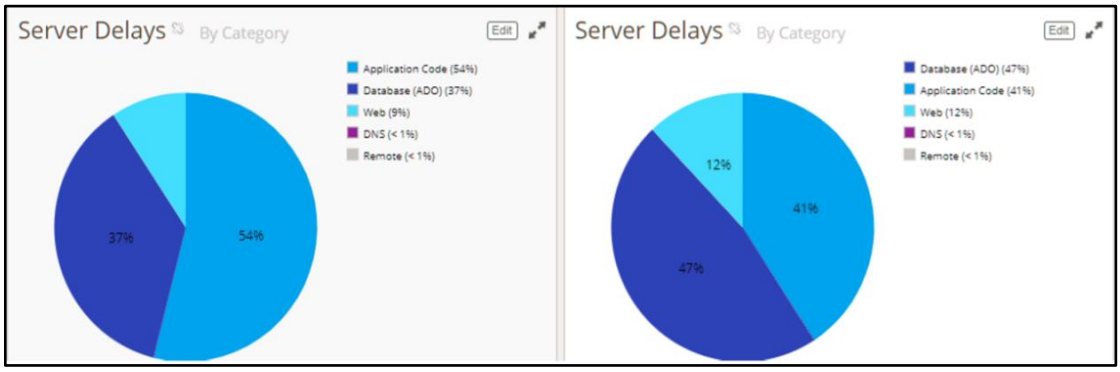
Şekil 3. 40 Ekran Bağımlılıkların İyileştirme Öncesi Analizi



Şekil 3. 41 Ekran Bağımlılıkların İyileştirme Sonrası Analizi

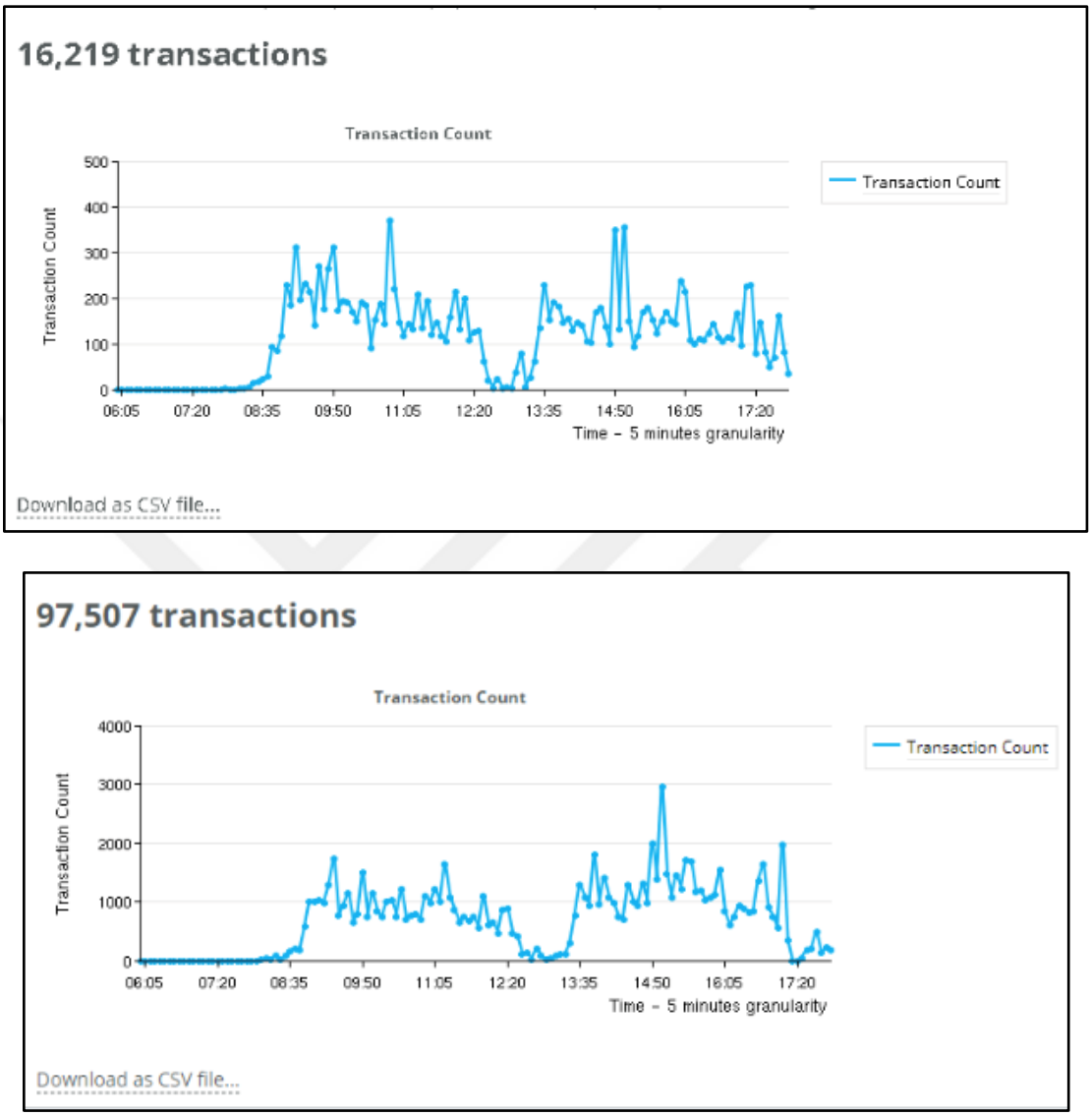
Şekil 3. 42 Response Sürelerinin Karşılaştırmalı Analizi

Tasarım deseni kullanılarak çevik yöntem ile yazılan kodun önceki yazılan koda göre performans kazancı skor kartın simüle isteği oluştururken geçen süreye göre 1456 sn iken 227 saniyeye düşmüştür.Application kod tüm kullanım içerisinde değişiklik öncesi %54 iken %41 e düşmüştür. Metot çağrımındaki response time değişiklik öncesi 37 milisaniye iken 29 milisaniyeye düşmüştür. Sadece transaction için elde edilen kazanç hesabı bir gün için yapıldığında yaklaşık 27 saatlik bir kazanç sağlanmıştır. Değişiklik öncesi istekler 1,5 sn çıkarken değişiklik sonrası 1 sn indiği görülmektedir.



Şekil 3. 43 Sunucuda Geçen Sürenin Karşılaştırmalı Analizi

Riverbed, serverda geçen toplam yükün ne kadarının veritabanında, kod içerisinde, dış kurumda ve network ağında geçtiğinin sürelerini hesaplamakta ve kullanıcıya grafiklerle aktarım yapmaktadır.

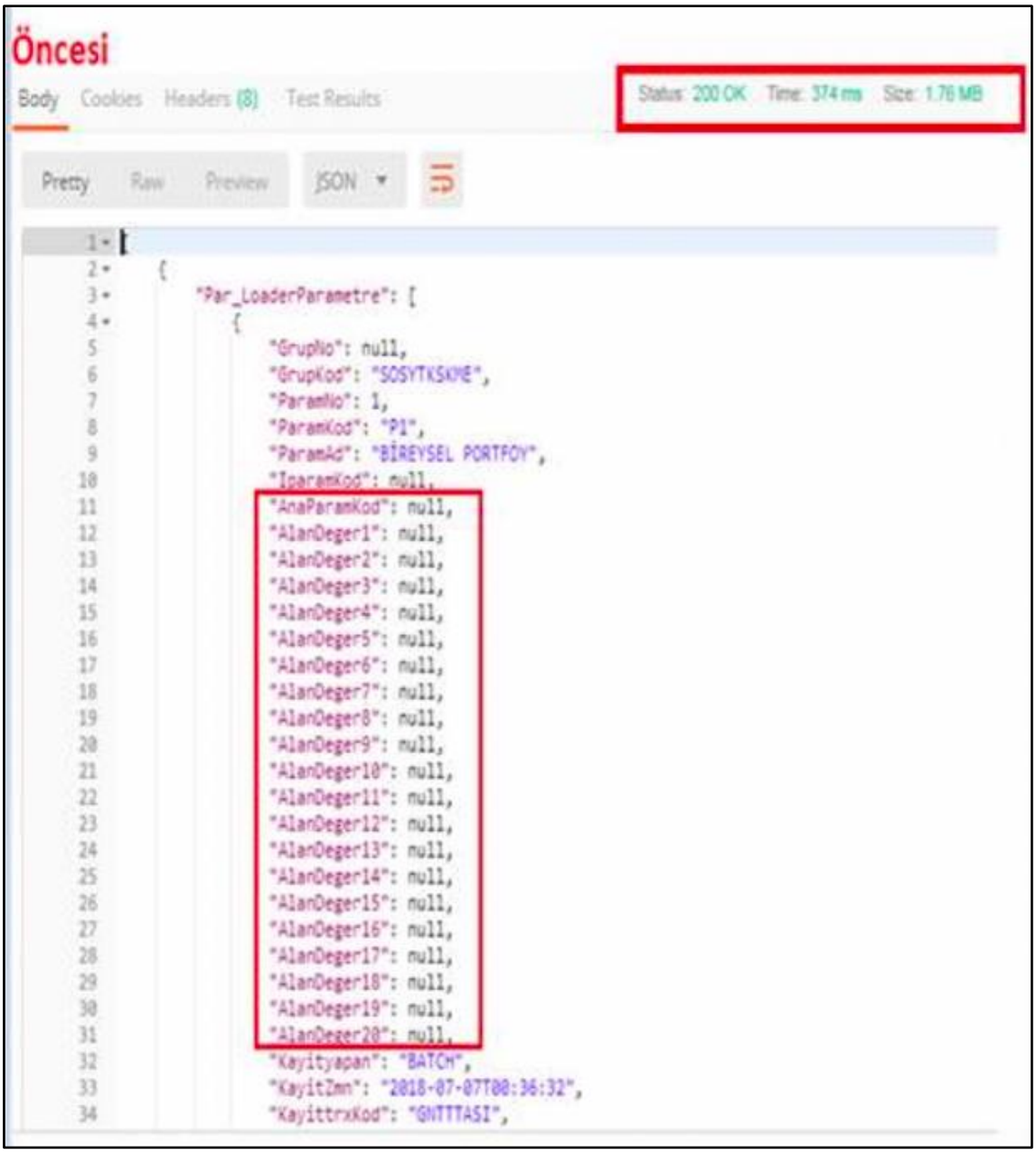


Şekil 3. 44 Request Çağrımlarının Riverbed Üzerinden Karşılaştırmalı Analizi

Değişiklik sonrası 3 sunucu üzerinde çağrım sayısı 97500 iken 16500'e düştüğü görülmektedir. Hesaplama yapıldığında aradaki farkın 81000 olduğu görülmekte ve toplam sunucu esas alındığında 1 milyon request kazanç sağlanmaktadır.

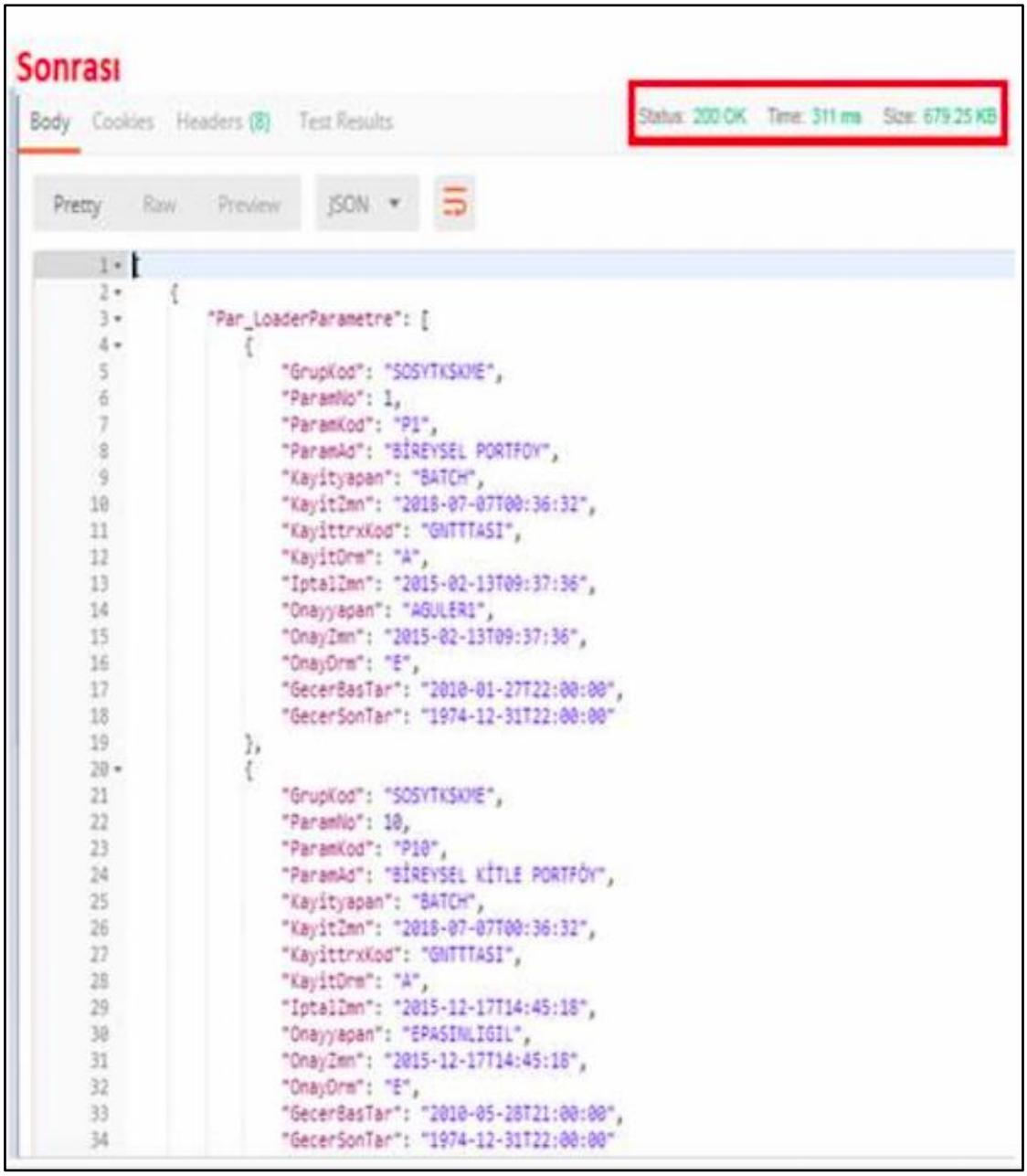
Aynı zamanda veriyi görselleştirerek analiz etme konusunda bir diğer kullanılan uygulamamız Kibana'dır. Uygulama sayesinde client IP ile, ilgili yaygınlaşan metot bazında log alınarak hata alan durumları irdeleyebilir, logların hangi zaman aralıklarında ne ölçüde kullanıldığını görebiliriz.

Aşağıdaki görselde esnek yapıda yazılan yazılımın skor kart yapısına BBE(Bireysel Borçluluk endeksi)kavramının dahil olması ile beraber hata alıp alma durumu irdelenmiştir.”



Şekil 3. 45 İyileştirme Öncesi Parametre Kullanım Boyutu

Şekil 3.45’de görüleceği üzere parametre yapısında Null olarak gelen değerler aslında kullanılmamasına rağmen server’a uğramakta ve datanın boyutunun bir hayli büyük olmasından dolayı sistemin performansını etkilemektedir.

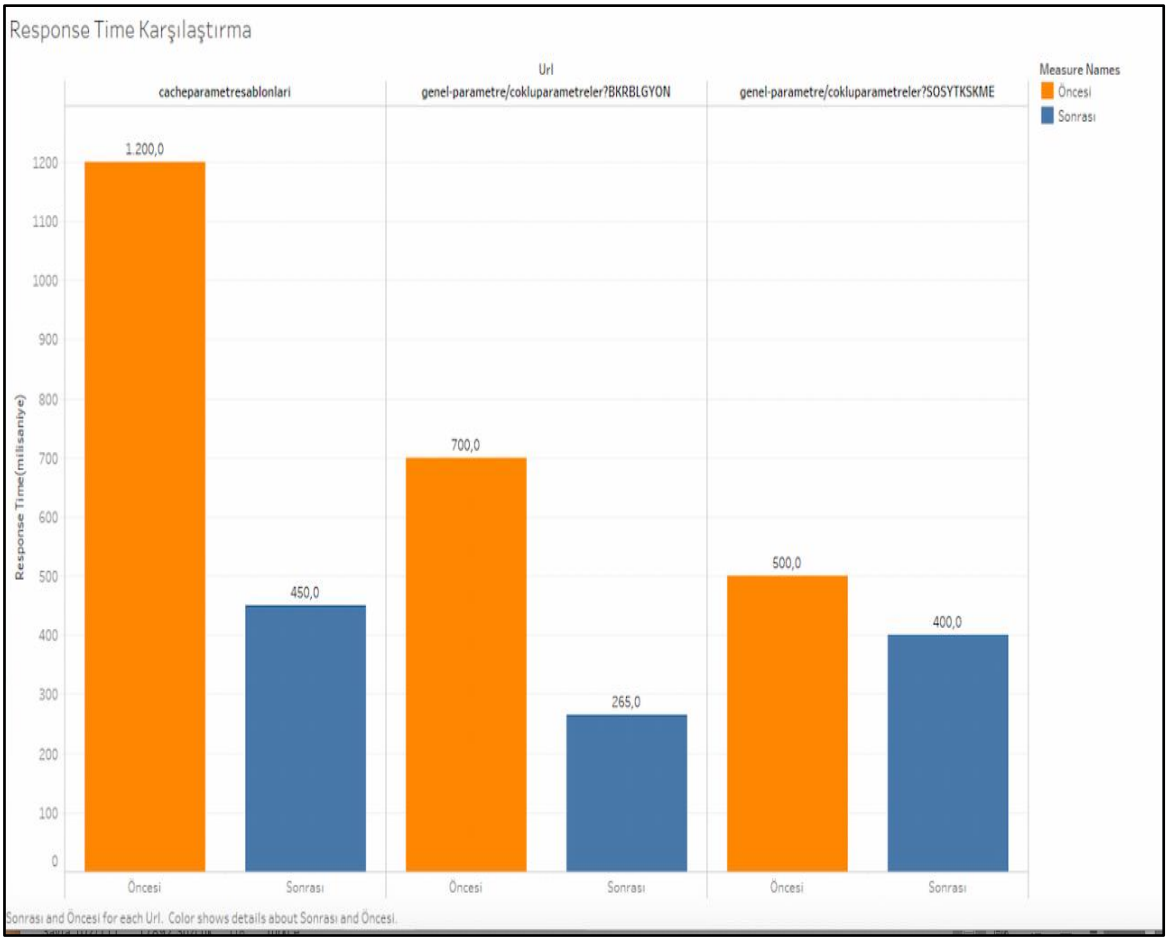


Şekil 3. 46 İyileştirme Sonrası Parametre Kullanım Boyutu

Şekil 3.46'da görüleceği üzere iyileştirme sonrası NULL olan değerlerin kod tarafında NULL ise server'a uğramadan değer dönülmemesi sağlanmış ve datanın daha hızlı client'e ulaşması sağlanmıştır. 3.47'de ilgili iyileştirme için NULL değerleri handling eden kod parçası gösterilmektedir.

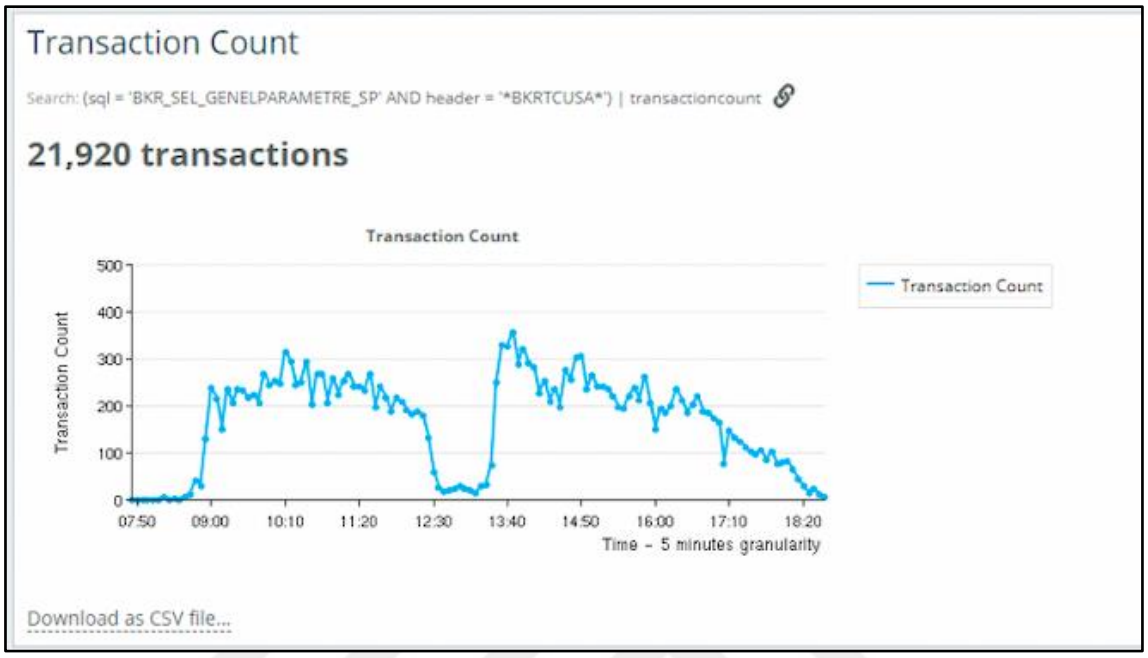
```
private string serializeContent(object content)
        {
            JsonSerializerSettings jsonSerSettings = new JsonSerializerSettings();
            bool isNullValueHandlingIgnore = m_Configuration.GetValue<bool>("NullValueHandlingIgnore", false);
            if (IsCamelCase)
            {
                jsonSerSettings.ContractResolver = new CamelCasePropertyNamesContractResolver();
                jsonSerSettings.NullValueHandling = isNullValueHandlingIgnore ? NullValueHandling.Ignore : NullValueHandling.Include;
                return JsonConvert.SerializeObject(content, jsonSerSettings);
            }
            else
            {
                jsonSerSettings.NullValueHandling = isNullValueHandlingIgnore ? NullValueHandling.Ignore : NullValueHandling.Include;
                return JsonConvert.SerializeObject(content, jsonSerSettings);
            }
        }
```

Şekil 3. 47 Handling Edilen Kod

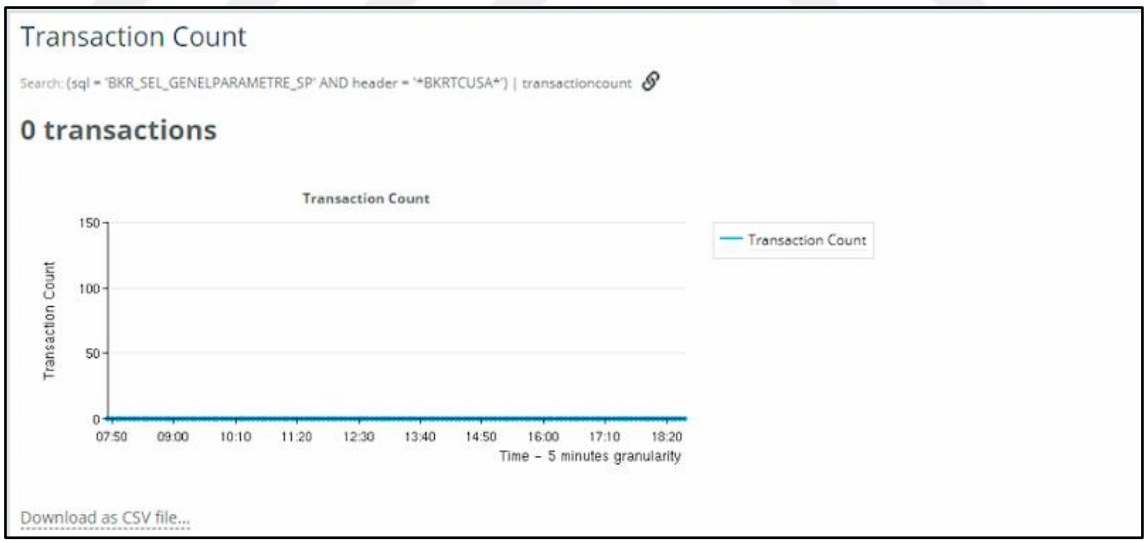


Şekil 3. 48 İyileştirme Öncesi ve Sonrası Url Response Süre Karşılaştırması

Yapılan fiddler incelemelerinde iş sürecinde kullanılmayan parametrelerin kod içerisinde kaldığı, hatta birden fazla Wall isteği ürettiği görülmüştür. Bu amaçla ilgili parametreler belirlenip kod içerisinden kaldırılması sağlanmıştır.



Şekil 3. 49 İyileştirme Öncesi Transaction Sayısı



Şekil 3. 50 İyileştirme Öncesi Transaction Sayısı

Şekil 3.49 ve 3.50'da kullanımı kaldırılan parametrelerin sistemde ürettiği işlem sayısı görülmektedir.

BÖLÜM 4

SONUÇ VE ÖNERİLER

Bu çalışmada, bir bankanın bireysel müşteri segment datası üzerinden geçmiş dönem kredi başvurularının skor puanı hesaplanmasında esas alınan, kredi değerlendirme kriterleri üzerinden yapılan değişikliklerin simülasyonu ve bu simülasyon sonucunda kredinin kabul edilmesi ya da reddedilmesi kararının verilmesine yönelik uygulanan kredi skorlama sisteminin iş analitiği açısından geçirdiği süreç ele alınmıştır. Değişikliğe uğrayan veri setinin bir bankanın gerçek veri kümesinde simüle edilmesi, anlamlı ve doğru bilgiye sadece önseziyle ulaşarak karar almayı riskli hale getiren bankanın temel iş metriklerini iyileştirmesine olanak sağlamıştır. Elde edilen sonuçlar finansal kuruşların daha doğru karar vermesine olanak sağlamakta ve kredi riskini tahminlemesiyle birlikte karlılık oranlarının artmasına imkan vermektedir. Bir başka açıdan analiz desenleri sayesinde, analiz riski düşürülerek, daha kısa sürede daha güvenilir yazılım ihtiyaçları ortaya çıkmaktadır. Analiz desenleri kullanan geliştiriciler, geliştirme zamanının düştüğünü, geliştirme kolaylığını ve daha düşük bakım maliyeti sağladığını tecrübe etmektedirler.

Çalışmada çevik yöntem ile tasarım deseni kullanılarak yazılan kodun performans ölçümü ile waterfall yöntem ile yazılan kodun performans ölçümü fiddler, kibana, azure, riverbed gibi uygulamalar esas alınarak karşılaştırmalı olarak incelenmiştir.

İncelemeler sırasında fiddler kurulumu yapılarak istek ve cevap süreleri karşılaştırılmış ve çevik yöntem ile tasarım deseni kullanılarak yazılan kodun yapılan işlem sayısının fazla olmasına rağmen işlem başına düşen zamanın daha performanslı olduğu ve işlemlere çok kısa zamanda cevap vermediği sonucuna varılmıştır.

Tasarım deseni kullanılarak yazılan kodun ve kullanılmadan yazılan kodun peformans değerlendirmesinde; farklı senaryolarda en çok istek yapan prosedürün, en çok işlem yapan ekranın, işlemciye ulaşana kadar geçen maksimum sürenin aşama aşama değerlendirmesi yapılarak iki yazılımın karşılaştırılması sağlamıştır. Kullanılan verilerin analiz incelemelerinde tasarım deseni kullanılarak yazılan kodun performans ölçümlerinin daha hızlı olduğu sonucu elde edilmiştir.

Süreç içerisinde kibana kullanılarak hata alan durumlar irdelenmiş ve iteraktif sonuç alınmıştır. Yazılan her iki kodun işlemci - yük dağıtıcı arasında geçen süre ile yük dağıtıcı - server arasında geçen süre riverbed uygulaması ile incelenmiş böylelikle yapılan işlemlerin network, server, veri tabanı, web servis üzerinde geçtiği sürenin performansı raporlanabilmiştir. Bu sayede uygulamanın çalışma performansı, yazılımın güvenliği, aynı anda çok fazla kullanıcı tarafından kullanıldığındaki kodun performansı ve ileride yapılacak değişikliklerin yazılımın esnek bir yapıya sahip olması ile kolay adapte olabileceği görülmüştür. Yapılan çalışma sonucunda geliştirilen yazılım mimari standartlara uygun, performansı yüksek ve gereksiz kodlamalardan arınmıştır.

Ayrıca zaman içerisinde yeni gereksinimler söz konusu olduğunda, çevik yöntem ile tasarım deseni kullanılan yapıda iş bağımlılıkları arasındaki ilişki net bir şekilde tanımlandığı için sürece dahil olması kolaylaşmıştır. Zaman içerisinde skor kart mantığına Bireysel Borçluluk Endeksi(BBE) kavramının kazandırılması bu sayede kolayca gerçekleştirilmiştir.Bu uygulamada çevik yöntem ile tasarım deseni kullanılarak geliştirilen yazılım kodunun ilerde yapılabilecek değişikliklere açık olduğu ve kod üzerinde yapılacak değişikliklerin yeni entegrasyonların yazılımına kolay uygulanabildiğini, performans ve hata tespitinin performans değerlendirme uygulamaları sayesinde kolayca raporlanabildiği gösterilmiştir.

[1] Fowler, M. (1997). Analysis Patterns: Reusable Object Models, Second Edition, Addison-Wesley, United States.

[2] Stoecklin, S. ve Allen, C. (2000). "Implementing Fowler's Analysis Validator Pattern in Java", Java Developer, 5: 9.

[3] Vlissides, J. (1997). "Patterns: The Top Ten Misconceptions", Object Magazine Online.

[4] Bobkowska, A. ve Grabowski J.(2009). "The Role of Analysis Patterns in System Analysis", Proceedings of EuroPlop, Gdansk University of Technology, Poland.

[5] Sesera, L.(2008). "Fundamental Banking Patterns", Proceedings of Plop, Faculty of Informatics and Information Technologies, Slovakia.

[6] Sesera, L.(2010). "Applying Fundamental Banking Patterns: Stories and Pattern Sequences", Proceedings of EuroPlop, Proceedings of the 15th European Conference on Pattern Languages of Programs, USA.

[7] Kazan,E. (2015). "Tasarım deseni kullanılarak geliştirilen yazılım ile kullanılmadan geliştirilen yazılımın performans analizi", Yüksek Lisans Tezi, Turgut Özal Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

[8] Al-Ahmad, W. (2006) "Object-Oriented Design Patterns for Detailed Design", Journal of Object Technology, 5: 155-169.

[9] Shi, N. ve Olsson, Ronald A. (2006). "Reverse Engineering of Design Patterns from Java Source Code", Automated Software Engineering, 21: 123-134.

[10] Mani, N., Dorina , C. ve Woodside , M. (2011). "Studying the Impact of Design Patterns on the Performance Analysis of Service Oriented Architecture", Software Engineering and Advanced Applications, 37: 12-19.

[11] Schmidt, Douglas C. (1996). "Experience Using Design Patterns to Develop Reuseable Object-Oriented Communication Software",Washington University.

[12] Koçyiğit, A ve Doğan, D.Y. (2014). “Bankacılık Kredi Alanı İçin Analiz Desenleri”, VIII. Ulusal Yazılım Mühendisliği Sempozyumu(UYMS), 8-10 Eylül 2014, KKTC, 511-522.

[13] Kalay, A. (2009). “Statik Kod Analizinin Yazılım Geliştirme Sürecindeki Yeri ve Yazılım Kalitesine Etkisi”, IV. Ulusal Yazılım Mühendisliği Sempozyumu, İstanbul, 211-218.

[14] Yıldız,H. (2019). “Finans Sektöründe Veri Madenciliği Kredi Skorlama”, Yüksek Lisans Tezi, Milli Savunma Üniversitesi Hezarfen Havacılık ve Uzay Teknolojileri Enstitüsü, İstanbul.

[15] Donel, B. (2012). “Yapay Sinir Ağları Yöntemi İle Kredi Skorlama”, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

[16] Soydaner, D. (2014).”Hibrit Yapay Sinir Ağları İle Kredi Skorlama”, Yüksek Lisans Tezi, Mimar Sinan Üniversitesi Fen Bilimleri Enstitüsü,İstanbul.

[17] http://archive.ics.uci.edu/ml/datasets/Statlog+(German+Credit+Data),UCI, Machine Learning Repository, Erişim Tarihi: 1 Haziran 2019.

[18] West, D. (2000). “Neural network credit scoring models”, Computers & Operations Research, 27: 1131-1152.

[19] Sönmez,F.,(2015). “Kredi Skorunun Belirlenmesinde Yapay Sinir Ağları ve Karar Ağaçlarının Kullanımı: Bir Model Önerisi”, ABMYO Dergisi, 40.

[20] Kasapoğlu,B. (2009). “Kredi Riskinin Hesaplanmasında Skorlama Yaklaşımı”, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Sosyal Bilimler Enstitüsü, İstanbul.

[21] Turangil, N.(2006). “Kredi skorlamada kullanılan yöntemler ve uygulamaları”,Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü,İstanbul.

[22] Karakuş,K. (2009). “Veri Madenciliği Teknikleri İle Mobil Telekom Sektöründe Müşterilerin Kredi Skorlamasına İlişkin İstatistiksel Bir Analiz“, Yüksek Lisans Tezi, Marmara Üniversitesi Fen Bilimleri Ensitüsü, İstanbul.

[23] ELİUZ, E.,(2009). “Tüketici Kredilerinde Risk Yönetimi ve Bir Skorkart Modeli Önerisi”, Yayınlanmış Yüksek Lisans Tezi, Marmara Üniversitesi, İstanbul, s.3.

[24] Oyakbank Eğitim Yayınları, 2005:22.

[25] Türkiye Bankalar Birliği Yayınları, 1999

[26] Lawrence and Solomon 2002.

[27] Karakaya, A.N. (2010). “Kredi Skorlama”, Yüksek Lisans Tezi, Selçuk Üniversitesi Fen Bilimleri Enstitüsü, Konya.

[28] Anderson, R.(2007). “The Credit Scoring Toolkit” , Oxford University Press, 1st Edition, U.K.

[29] Mramor, D., ve Zupan, J. (2009). “Consumer Credit Scoring Models With Limited Data”, Expert Systems with Applications,36: 4736-4744.

[30] Lee, T.S., Chiu, C.C., Lu, C.J. ve Chen, I.F. (2002). “Credit Scoring Using The Hybrid Neural Discriminant Technique”, Expert Systems with Applications, 23: 245-254.

[31] Çinko, M. (2010). “Kredi Kartı Değerlendirme Tekniklerinin Karşılaştırılması”, 14. Ulusal Finans Sempozyumu, Selçuk Üniversitesi İktisadi ve İdari Bilimler Fakültesi, Konya.

[32] TATLIDİL, H. (2002). “Uygulamalı Çok Değişkenli İstatistiksel Analiz”, Ziraat Matbaacılık, Ankara.

[33] Cinemre, N. (2004). “Yöneylem Araştırması”, Beta Basım Yayım Dağıtım A.Ş., İstanbul.

[34] Cinemre, N. (2004). “Doğrusal Programlama”, Beta Basım Yayım Dağıtım A.Ş., İstanbul.

[35] Thomas, L.C. (2000). “A Survey of Credit and Behavioural Scoring: Forecasting Financial Risk of Lending to Customers”, International Journal of Forecasting, 16: 149-172.

[36] Mangasarian, O.L. (1965). “Linear and Nonlinear Separation of Patterns by Linear Programming”, Operations Research, 13: 444-452.

[37] Özer, B. ve Kuşlu, M. (2012). “Yerel Yönetimlerdeki Yönetim Bilgi Sistemleri ve Karar Destek Süreçleri: Manisa Belediyesi Örneği”, Celal Bayar Üniversitesi Sosyal Bilimler Dergisi, 10: 1.

[38] Angelini, E., Tollo, G. ve Roli, A. (2008). “A Neural Network Approach for Credit Risk Evaluation”, The Quarterly Review of Economics and Finance, 48:733-755.

[39] Şen, Z. (2004). “Yapay Sinir Ağları İlkeleri”, Su Vakfı Yayınları, İstanbul.

[40] Malhotra, R. ve Malhotra, D.K. (2003). “Evaluating Consumer Loans Using Neural Networks”, Omega (The International Journal of Management Science), 31: 83-96.

[41] Çil, İbrahim. “Problem Tanımlama ve Çözme Yaklaşımları - SABİS”. http://content.lms. sabis. sakarya.edu.tr/Uploads/49866/38789/10.hafta.pdf, Erişim Tarihi: 1 Haziran 2019.

[41] Soegaard, M. (2002). “The Basics of User Experience Design: A UX Design Book by the Interaction Design Foundation”, https://www.interaction-design.org/literature/topics/ ux-design, Erişim Tarihi: 1 Haziran 2019.

[42] Royzen, Z. (1993). “Application TRIZ in Value Management and Quality Improvement”, International Conference of the Society Of American Value Engineers, Florida.

[43] Şener. S.D. (2006). “Trız: Yaratıcı Problem Çözme Teorisi ve Diğer Problem Çözme Yöntemleriyle Karşılaştırma”, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

[44] Altshuller, G. (2000). “The Innovation Algorithm: TRIZ, Systematic Innovation and Technical Creativity”, Technical Innovation Center, Inc., Worcester.

[45] Yıldız, E. (2004). “Yaratıcı Problem Çözme Teorisi ve Uygulamaları”, Yüksek Lisans Tezi, İ.T.Ü. Fen Bilimleri Enstitüsü, İstanbul.

[46] Bozkurt, R. (2003). “Kalite İyileştirme Araç ve Yöntemleri”, Milli Prodüktivite Merkezi Yayınları.

[47] Köksal, G. (2001). “Problem Çözme Teknikleri”, Orta Doğu Teknik Üniversitesi, Eğitimde Toplam Kalite Yönetimi Semineri, Yalova.

[48] Binboğa B. (2015). “İnternetten Alışverişte Ürün İade Sürecinde Lojistik Destek Sağlayacak Şirket Seçimi: Analitik Hiyerarşi Süreci Metodu İle Uygulama”, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü.

[49] Şimşek, M. (2001). “Toplam Kalite Yönetimi”, Alfa Basım Yayım Dağıtım Ltd. Şti., İstanbul.

[50] George, M. (2002). “Lean Six Sigma: Combining Six Sigma Quality with Lean Speed”, McGraw-Hill.

[51] George, M., Rowlands, D. ve Kastle, B. (2005). “Yalın Altı Sigma Nedir?”, SPAC Yayınları Ankara.

[52] Ateş, G.S. (2008), “Alti Sigma Yaklaşimi ve Bir Bankada Müşteri Memnuniyetini Artirmaya Yönelik Alti Sigma Uygulamalari”, Yüksek Lisans Tezi, Selçuk Üniversitesi Sosyal Bilimler Enstitüsü, Konya.

[53] Şener, S.D. (2006). “TRIZ: Yaratıcı Problem Çözme Teorisi ve Diğer Problem Çözme Yöntemleriyle Karşılaştırma”, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

[54] Yaralıoğlu, K. “İşletme Sorunlarının Çözümünde Yaratıcı Sorun Çözme Teorisi.”, 1.Ulusal Kalite Fonksiyon Göçerimi Sempozyumu, http://www.deu.edu.tr/ userweb/k.yaralioglu/dosyalar/TRIZ.doc, Erişim Tarihi: 1 Haziran 2019.

[55] Aydoğan, E.K., Gencer, C. ve Akbulut, S. (2008). “Veri Madenciliği Teknikleri İle Bir Kozmetik Markanın Ayrılan Müşteri Analizi ve Müşteri Bölümlenmesi”, Mühendislik ve Fen Bilimleri Dergisi, 26: 46.

[56] Enes,A.(2018).“Nedir Bu Tasarım Desenleri?(Design Patterns)”, http://enesaysan.com/ software/c-tasarim-desenleri-design-patterns/, Erişim Tarihi: 1 Haziran 2019.

[57] Boutquin, P., Poremsky, D. And Slovak, K., (2000). “Beginning Visual Basic 6 Application Development, UK, Wrox Press.

[58] Schwalbe, K., (2000). “Information Technology Project Management”, Course Technology Thomson Learning, Canada.

[59] Rehber,D. “Yazılım Projelerinde Başarısızlık”, http://www.emo.org.tr/ekler/ d2129f927262c5b_ek.pdf, Ankara, Erişim Tarihi: 1 Haziran 2019.

[60] Yılmaz, C. (2016).“Değişiklik Maliyeti”, http://www.yilmazcihan.com/degisiklik-maliyeti/, Erişim Tarihi: 1 Haziran 2019.

[61] Güzel, A.N.S., Yazıcıoğlu, O., Borat, O. (2016). “Bir Şirket İçin İnsan Kaynakları İçin Uygulama”, Marmara Fen Bilimleri Dergisi, https://dergipark.org.tr /download/article-file/274204, İstanbul, Erişim Tarihi: 1 Haziran 2019.

[62] Ünsal, S. (2014). “Dijital Girişimci İçin Ürün Yönetimi”, http://paradigmayayinlari.com/wp-content/uploads/2017/08/ DijitalGirisimcininElKitabı.pdf, Erişim Tarihi: 1 Haziran 2019.

[63] Kniberg, H. (2016). http://blog.crisp.se/author/henrikkniberg, Erişim Tarihi: 1 Haziran 2019.

[64] Fowler, M. (2003). UML Distilled: A Brief Guide to the Standard Object Modeling Language (3rd Edition) . M. Fowler içinde, UML Distilled.

[65] Güngören, B. (2009).“UML İle Nesne Tabanlı Çözümleme ve Tasarım”, 1.Baskı, Seçkin Yayıncılık, İstanbul.

[66] Yüregir, O.H. (2001). “Bilişimde Sistem Analizi ve Tasarımı”, Nobel Kitabevi, Adana.

[67] Birol, A., (2016). “Kredi Risk Yönetimi ve Modelleme”, Yüksek Lisans Tezi, İstanbul Ticaret Üniversitesi Fen bilimleri Enstitüsü, İstanbul.

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Çisem ÖZKAN

Doğum Tarihi ve Yeri : 14.06.1989, İstanbul

Yabancı Dili : İngilizce

E-posta : aksu.cisem89@gmail.com

ÖĞRENİM DURUMU

Derece	Alan	Okul/Üniversite	Mezuniyet Yılı
3,5	Matematik Mühendisliği	Yıldız Teknik Üniversitesi	2013
3.89	Sayısal	Bakırköy Anadolu Lisesi	2007

İŞ TECRÜBESİ

Yıl	Firma/Kurum	Görevi
2018	Ziraat Teknoloji A.Ş.	Kıdemli İş Analisti Uzmanı
2015	Vakıfbank EBİS A.Ş.	İş Analisti Uzmanı
2013	Anadolubank A.Ş.	Sistem Geliştirme Uzmanı

YAYINLARI

Bildiri

[1] ÖZKAN,Ç., ŞAHİNTÜRK, H. (2017). “Bankacılık Uygulaması Olan Kredi Skor Puanlama Sisteminin İş Analitiği Teknikleri Kullanılarak İncelenmesi”, 2. Uluslararası Mühendislik ve Tasarım Kongresi, 12-13 Mayıs 2017, Sayfa: 684-685.