T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ BİLİM DALI

BANKACILIKTA BİR UZMAN SİSTEM TASARIMI

Yüksek Lisans Tezi

Tezi Hazırlayan: **Gülşen COŞKUN**

İstanbul, 2011

T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ BİLİM DALI

BANKACILIKTA BİR UZMAN SİSTEM TASARIMI

Yüksek Lisans Tezi

Tezi Hazırlayan:
Gülşen COŞKUN
Öğrenci No:
080820009

Danışman:
Prof.Dr.M. Yahya Karslıgil

İstanbul, 2011

YEMİN METNİ

Yüksek lisans tezi olarak sunduğum "Bankacılıkta Bir Uzman Sistem Tasarımı" başlıklı bu çalışmanın, bilimsel ahlak ve geleneklere uygun şekilde tarafımdan yazıldığını, yararlandığım eserlerin tamamının kaynaklarda gösterildiğini ve çalışmamın içinde kullanıldıkları her yerde bunlara atıf yapıldığını belirtir ve bunu onurumla doğrularım. 21/01/2011

Gülşen Coşkun

BANKACILIKTA BİR UZMAN SİSTEM TASARIMI

Tezi Hazırlayan: Gülşen Coşkun

ÖZET

Yazılım projelerinin başarılarını etkileyen birçok kriter vardır. Bunlardan ilk akla gelenler; yazılım ekibinin bilgi ve becerisi, iş kapsam ve tanımının tam olarak yapılması ve seçilen yazılım geliştirme ortamının, üretilmek istenen yazılım konusundaki yeterliliğidir. Başarıyı sağlayabilmek için bu kriterler üzerinde, istatistiklere dayalı birçok inceleme yapılmaktadır. Yapılan bu çalışmada, "bankacılık projelerinde başarı oranı daha fazla nasıl yükseltilebilir" sorusunun cevapları, sistem gelişimi sırasında uygulanmıştır.

Projelerde, test aşamasında bulunan hataların maliyet oranının, bitmiş projelerin, uygulama aşamasında çıkan hataların maliyet oranından daha düşük olduğu görülmüştür. Dolayısıyla test işlemlerini yürüten, test ekibine verilen önem artmıştır. Bu konudan yola çıkarak, iş analizi ve test ekiplerinin hazırladığı senaryoların, sistemi uçtan uca kontrol edebilecek derinlikte olması, çeşitli kombinasyonların kontrol edilmesi gerekmektedir. Yoğun çalışma hayatında, test senaryolarının hazırlanması sırasında, detay senaryolar bazen gözden kaçabilmekte, akla gelmemektedir. Eğer test ekibi de konu hakkında detaylı bilgiye sahip değilse, oluşabilecek her hata, test sürecinde gözden kaçtığı için, maliyet oranında artış meydana getirmekte ve yazılıma olan güven azalmaktadır.

Geliştirilen sistem, iş analizi ve test ekiplerinin hazırladığı test senaryolarının eksiksiz olmasını sağlamaktadır. Böylece yazılımda çıkabilecek her hata kontrol edilmiş olup, müşteri kullanmaya başladığında hata almaması sağlanacaktır. Teorik olarak analizi yapılan ve geliştirilen sistemin uygulaması, örnek olarak alınan kredi kartları projelerindeki, testleri yapılmış, sorunsuz çalışan bir modülün test senaryoları oluşturularak gerçekleştirilmiştir. Modülün, analiz dokümanı kullanılarak, ekranda kontrol edilmesi istenen her parametre belirlenmiştir. Geliştirilen sisteme girişi yapılan parametreler ile manuel, otomatik ve analiz dokümanı kullanılarak test senaryoları oluşturulmaktadır. Excel dokümanı olarak oluşturulan test senaryoları,

test edilecek sistemi, baştan sona kontrol edebilecek niteliktedir. Testler sırasında kullanılacak test aracı varsa oluşturulan doküman araca yüklenebilir, yoksa doküman üzerinden senaryolar ile sistem test edilebilir.

Anahtar Kelimeler: Yazılım testi, sistem testi, test senaryolarını hazırlama, bankacılıkta uzman sistem

DESIGN OF AN EXPERT SYSTEM AT BANKING

Presented by: Gülşen Coşkun

ABSTRACT

There are several criteria that affect the success of software projects. The first of these that come to mind are the software team konwledge and skills, ensuring that the work definition and ambit are completely indicated, the adequacy about software which want to be produced of the selected development media of. To ensure the success, based on statistics lots of study are made on these criteria. In this study, the answer of "How can success rate of credit card projects be upgraded" question was tried to be applied during the example system development.

That has been seen at the projects that the cost ratio of errors in the testing phase was lower than the cost ratio of errors in the implementation phase at the completed projects. Therefore, the importance of the quality assurance team who carry out testing procedures has increased. Based on this topics, The scenarios which were prapared by business analysis and quality assurance teams must be a depth that be able to controll the system entirely and the various combinations must be checked. In the intensive work life, during preparation of the test scenarios, detail scenarios sometimes can't be seen or don't come to mind. If the test team does not have detailed knowledge of this subject, Due not to be seen during the testing process, every errors that may occur cause an increase of cost rate and the reliability of software declines.

The developed system provides that the scenarios, which are prepared by business analysis and quality assurance teams, are compelete. Thus, every possible mistakes in software are tested , the customer will not receive an error when he/she uses. For example, a seamlessly running module from member campany modules in the credit cards projects has been considered. Each parameter which was wanted to be checked on the screen have been determined using analysis document of the module. Tests scenarios have been created using manual, automatic and analyze the documents with input parameters of developed system. Test scenarios wich was

created as excel document can control the system that will be tested entirely. If there is a test tool, the document which was created can be downloaded in it, if not the system can be tested with scenarios through the document.

Keywords: Software testing, system testing, to prepare the test scenarios, an expert system at banking

İÇİNDEKİLER

TABLOLAR LİSTESİ

ŞEKİLLER LİSTESİ

KISALTMALAR

BKM	Bankalararası Kart Merkezi
BM	Boyer Moore
EMV	Europay, MasterCard, VISA
FMEA	Failure Mode Effect Analysis
IP	Internet Protocol
KMP	Knuth Morris Prat
POS	Point of sale
RPN	Risk Priority Number
SQL	Structured Query Language
YD	Yurt Dışı
YI	Yurt İçi
YSA	Yapay Sinir Ağları

GİRİŞ

Bilgi teknolojilerinin gelişmesiyle pek çok alanda işleyişi kolaylaştıran yazılımlar kullanılmaktadır. Bu yazılım projelerinin başarılarını etkileyen birçok kriter vardır. Bunlardan ilk akla gelenler; yazılım ekibinin bilgi ve becerisi, iş kapsam ve tanımının tam olarak yapılıp yapılmadığı ve seçilen yazılım geliştirme ortamının, üretilmek istenen yazılım konusundaki yeterliliğidir. Bu kriterler konusunda istatistiklere dayalı birçok inceleme yapılmaktadır. Yapılan tüm incelemelerin ana teması tek bir konu üzerindedir: Yazılımda başarı nedir? Zaman planı, maliyet ve kalite konularında, proje beklenenleri karşılıyor ise yazılımda başarı sağlanmış kabul edilebilir mi?

Zaman planı; yazılımın çeşitli hedef noktalarına belirlenen zamanda varılmasını ve hedefleri yerine getirilmesini kapsar. Bir yazılım projesinde bulunması beklenen analiz kapanış, tasarım kapanış, kod geliştirme kapanış gibi hedeflere vaktinde ulaşılması örnek oluşturmaktadır. Bunun sağlanabilmesi için hedef noktalarına erişen proje adımlarının her birinin vaktinde ya da daha erken bitirilmesi gerekmektedir. Bu çerçevede zaman planına uymayı hedefleyen bir proje ekibinin odaklanması gereken, hedefin kendisi ve bu hedefe ulaşmak için gereken her adımın zamanında bitirilmesidir[1].

Maliyet; proje tamamlandığında öngörülen bütçe sınırları içinde kalmak üzere bu hedefe varmaktır. Proje tamamlanma noktasına gelmeden önce, varılması gereken hedefler vardır. Bu hedeflere erişmek için oluşturulan bütçelerin sağlanması ve bunun takibi, tüm projenin bütçe hedefini tutturmasında önemli bir rol oynayacaktır[1].

Zaman ve maliyetin dışında yazılımda kalite; hatasız, planlanan bütçe ile zamanında bitirilip dağıtılabilen, ihtiyaçları ve beklentileri karşılayabilen, sürdürülebilir özelliklere sahip yazılımlarla sağlanabilir. Kalite, kişilere göre değişebilen bir kavram olup, müşterisinin kim olduğuna ve tasarımda hedeflenen unsurlara bağlı olarak farklılıklar gösterebilmektedir. Doğal olarak, her kişinin kalite hakkında bireysel eğilimleri, tercihleri söz konusu olmasına karşın, kaliteyi ortaya koyan nesnel yöntemler, yansız değerlendirmeleri olanaklı kılmaktadır.

Kalite grubunun yazılım testlerinde karşılaştığı sorunlar hata olarak değerlendirilir. Kalitede başarıyı aradığımız zaman, istenenleri tamamıyla karşılamak ve ortaya çıkan sonucun, yazılımın müşterisi ve kullanım alanının kabul edilebilir bir hata düzeyinde teslim etmek olduğu görülmektedir. Yazılım endüstrisinde bu hatalar, genellikle beş sınıfa ayrılmaktadır:

- Kritik Önemli (Critical)
- Çok Önemli (High)
- Önemli (Medium)
- Az Önemli (Low)
- İyileştirme (Enhancement)

Yazılım kalite uzmanının tecrübesine, sistemi tanımasına bağlı olarak, yazılımı bir adım daha öne götüreceğini düşündüğü özellikleri, iyileştirme şeklinde yazılım ekibine bildirebilir. Böylece yazılım müşterisi için optimum fayda, memnuniyet sağlanmış olmaktadır. Projelerin kabul kriterleri, özellikle yazılacak kod satır sayısının büyüklüğüne bağlı olarak, yukarıdaki hata sayıları ile belirlenmektedir [1]. Araştırmaların sonucunda, başarı oranının yukarı çekilmesi ile ilgili yapılması gerekenler teorik olarak bilinmesine rağmen, uygulamada problemler yaşandığı için bu kadar düşük başarı oranları ile karşılaşılmaktadır.

Yapılan bu çalışmada, "bankacılık projelerinde başarı oranı daha fazla nasıl yükseltilebilir" sorusuna cevaplar aranmıştır. Yazılım projelerinde, test aşamasında bulunan hataların maliyet oranının, bitmiş projelerin uygulama aşamasında çıkan hataların maliyet oranından daha düşük olduğu görülmüştür. Dolayısıyla test işlemlerini yürüten kalite güvence ekibine verilen önem artmıştır. İş analizi ve test ekiplerinin hazırladığı test senaryolarının, sistemi uçtan uca kontrol edebilecek derinlikte olması, farklı durumların kontrol edilmesi gerekmektedir. Yoğun çalışma hayatında, test senaryolarının hazırlanması sırasında, detaylı senaryolar bazen gözden kaçabilmekte ve akla gelmemektedir. Eğer test ekibi de konu hakkında detaylı bilgiye sahip değilse, oluşabilecek her hata, test sürecinde gözden kaçtığı için, maliyet oranında artış meydana getirmekte ve yazılıma olan güven azalmaktadır. Geliştirilen sistem, iş analizi ve test ekiplerinin hazırladığı test senaryolarının eksiksiz olmasını sağlamaktadır. Böylece yazılımda çıkabilecek her hata kontrol edilmiş olup, müşteri kullanmaya başladığında hata almaması sağlanacaktır.

Teorik olarak analizi yapılan ve geliştirilen sistemin uygulaması, örnek olarak alınan kredi kartları projelerindeki, testleri yapılmış, sorunsuz çalışan bir modülün test senaryoları oluşturularak gerçekleştirilmiştir. Modülün, analiz dokümanı kullanılarak, ekranda kontrol edilmesi istenen her parametre belirlenmiş, bu parametreler ile geliştirilen sistemde manuel, otomatik ve analiz dokümanından test senaryoları oluşturulmuştur. Elde edilen test senaryolarının, testleri yapılacak ekranı baştan sona kontrol edebilecek derinlikte olduğu görülmüştür. Geliştirilen sisteme yüklenen veriler değiştirilerek, diğer sektörlerde kullanılan yazılım paketleri için de test senaryosu oluşturulabilir.

Tez kitabında; yapılan fizibilite çalışmaları, sistem analizi, sistem tasarımı ve varolan uygulamaların incelenmesine yer verilmiştir. Ekler bölümünde ise geliştirilen sistemin ihtiyacı olmayan, yazılım testi ve yazılım kalitesiyle ilgili bilgiler bulunmaktadır.

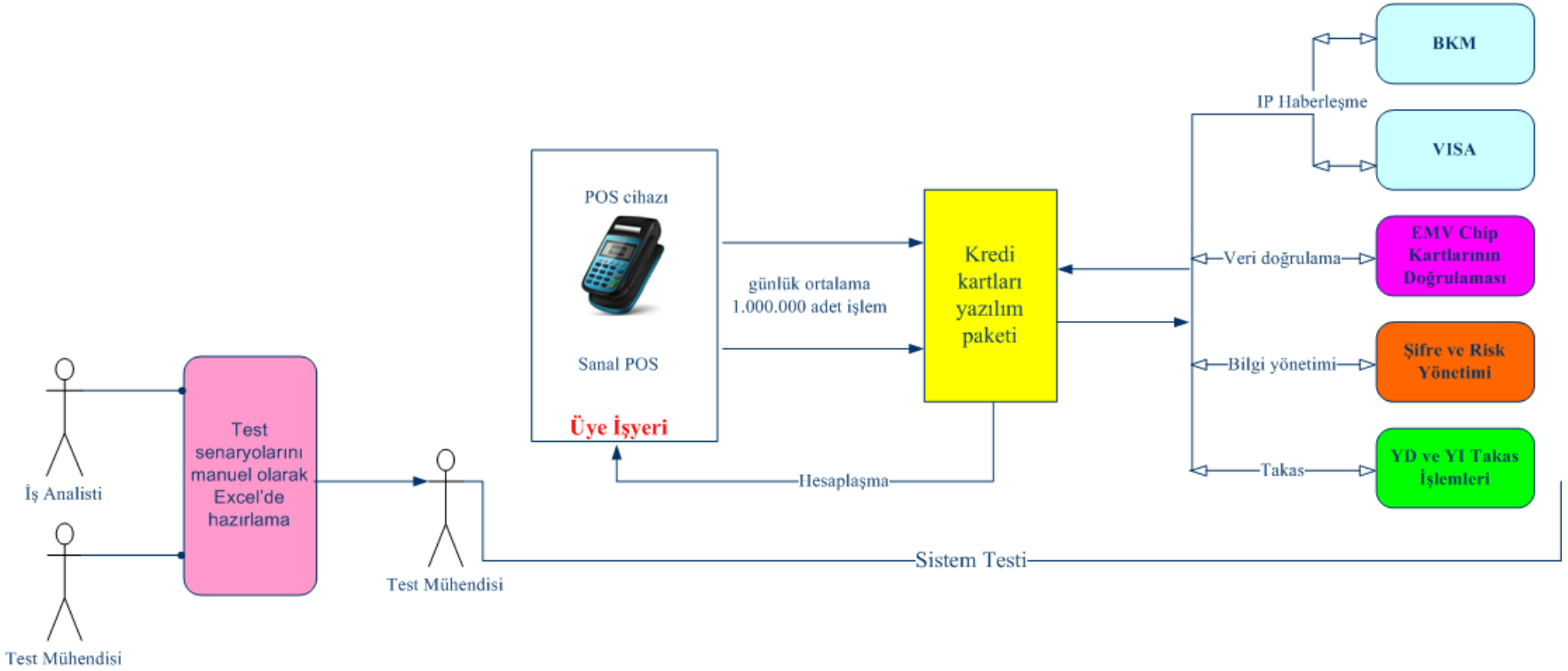
1. ÖN HAZIRLIK ÇALIŞMALARI

1.1 Varolan Sistemin Yapısı

Bu çalışmada, bankacılık yazılımlarının testlerinde kullanılabilir bir sistem geliştirilmiştir. Bankaların kullandığı yazılım paketleri, orta ve büyük ölçekli çok sayıda projeyi içermektedir.

Örneğin kredi kartları projelerinde, BKM (Bankalararası Kart Merkezi) ve VISA IP (Internet Protocol) haberleşmesi, günlük ortalama 1.000.000 adet işlem, bu yazılım paketleri üzerinden sağlanmaktadır. Ayrıca EMV (Europay, MasterCard, VISA) chip kartlarının doğrulaması, şifre ve risk yönetimi ve Mastercard takas işlemlerinin de yapılması, testleri yapılan uygulamaların, bir banka için ne derece önemli olduğunu göstermektedir. Böylesine büyük yazılımlarda çıkabilecek her hatanın maliyeti, bulunduğu aşamaya göre farklılık göstermektedir. Geliştirme işleminin erken aşamalarında yanlışları saptayarak ileri aşamalara yayılmasını önlemek, böylece zaman ve maliyetten tasarruf sağlamak çok önemlidir. Yapılabilecek küçük bir hata, büyük miktarlarda para kaybına yol açabilmektedir. Ayrıca üretilen yazılım yeterince test edilmeden müşterinin hizmetine sunulduğunda, olası yazılım hataları sonucunda itibar kaybedileceği gibi müşterilerine tazminat ödemek durumunda da kalınabilmektedir.

İş analistleri ve test ekipleri, sistem testinde kullanılacak olan test senaryolarını, analiz dokümanlarına göre genellikle Excel'de manuel olarak oluşturmaktadır. Varolan yazılımlarda test senaryolarının, geliştirilen uzman sistem tarafından oluşturulması, çıkabilecek hataları gözden kaçırma ihtimalini ortadan kaldırmaktadır. Varolan sistemlerin genel yapısı aşağıdaki şekilde gösterilmektedir.



Şekil 1.1 Varolan sistemin yapısı

1.2 Varolan Uygulamaların İncelenmesi

Test araçları; kullanıcının ihtiyaçları dokümante edebilmesini, planlamasını, tasarımı, kodlamayı ve testi karşılamaktadır. Kısacası geliştirme sürecinin her aşamasında aktif olarak kullanılabilmektedir [12]. Geliştirilen sisteme konu olan test aracı, test aşamasında yer almaktadır. Bu bölümde yazılım testinde, test adımlarının oluşturulmasında kullanılan araçlar incelenmiş olup, bilgilerine aşağıda yer verilmiştir. Yazılım sektöründe en çok kullanılan araçlar: HP Mercury; Quality Centre. IBM Rational Tools; Rational Quality Manager, Requisite Pro. Microsoft; Test Manager 2010.

İncelenen sistemler, test işlemlerinin tamamını kapsamaktadır. Test adımlarının manuel olarak oluşturulması, birim test seviyesinde otomatik test adımlarının oluşturulması ve işletilmesinin yanı sıra, kullanıcıların ortak çalışmasını sağlayan ek özellikleri de bulunmaktadır. Fakat analiz dokümanı kullanılarak, test adımlarının sistem tarafından otomatik olarak oluşturulması özelliği bulunmamaktadır. İncelemelerin ve ihtiyaçların sonucunda, yeni bir sistem geliştirilmesine karar verilmiştir.

1.2.1 HP Quality Centre

İstemci-Sunucu teknolojisine dayanan web tabanlı bir test yönetim aracıdır. Beş ana modülden oluşmaktadır. Yayın yönetimi; proje yöneticisi tarafından ürünün geliştirme aşamalarına göre teste başlamadan önce bültenlerin hazırlandığı bölümdür. Hangi sürümde hangi değişikliklerin ve geliştirmelerin yapılacağının öngörüldüğü modüldür.

Gereksinimler modülü; gereksinimlerin oluşturulduğu bölümdür. Birçok gereksinim toplama ve test aracı (Blueprint, Requisite Pro borland…) HP QC ile uyumludur. Kullanıcılar, gereksinimleri eklentiler aracılığıyla QC'ye aktarılabilir.

Test Planı modülü; test planlarının oluşturulmasında kullanılır. Test senaryoları oluşturulur ve güncellenir. Oluşturulan test senaryoları klasörler altında saklanır. Klasör mantığı projeye göre farklılıklar gösterebilir. Bazı projelerde aktörler bazında klasörler oluşturulurken, bazı projelerde ise fonksiyon bazlı ağaç yapısı

oluşturulabilir. Microsoft Excel tabloları ile oluşturulmuş senaryoların toplu olarak araca aktarımı gerçekleştirilebilir.

Test Laboratuarı modülünde; testler bir çevrimde gerçekleştirilebileceği gibi, birden fazla çevrimlerle de yapılabilir. Test senaryolarının gruplandırılarak çevrimlere dağıtılması test laboratuarı üzerinden gerçekleştirilir. Bu modülde sadece çevrim setleri oluşturulur. Test senaryoları ise test planından seçilerek ilgili setlere birer kopyalarının aktarılması ile oluşturulur.

Hatalar modülünde bütün hataların kaydedilmesi sağlanır. Hatanın durumu, tarihi, önem dereceleri gibi bir çok kritere göre filtrelenerek listeleme yapılabilir. Hataların yaşam döngüsü bu bölümden takip edilir.

1.2.2 IBM Rational RequisitePro

IBM Rational RequisitePro, müşteri gereksinimlerinin tanımlanmasını, diğer gereksinimlerle ya da diğer yazılım çıktıları (test senaryosu, model, kod vs.) arasındaki etki analizlerinin yapılmasını ve gereksinimlerin yönetilmesini sağlamaktadır [7].

Yaygın olarak kullanılan Microsoft Word belgelerinin kolaylığını, daha etkin gereksinim analizi ve sorgulaması için güçlü veritabanı yetenekleri ile birleştirir. Tüm takımların, gereksinimler üzerindeki değişimin etkisini takip edebilme olanağı sağlar. En güncel gereksinim bilgilerinden haberdar olunmasını sağlayarak doğru ve tutarlı gereksinimler üzerinde çalışmaya imkan verirken, dağınık ekipler için de web tabanlı erişim sağlamaktadır [7].

1.2.3 Rational Quality Manager

Rational Quality Manager, kalite güvence ve test çalışmalarını her açıdan izlenmesinde yardımcı olan bir araçtır. Takımların bilgileri kesintisiz bir şekilde paylaşmalarına, proje zamanlamalarını hızlandırmak için otomasyonu kullanmalarına ve bilinçli yayın kararları almak üzere proje metrikleriyle ilgili raporlar hazırlamalarına olanak tanıyarak, işbirliği içinde çalışmalarına yardımcı olmaktadır [7]. Rollerin, süreçlerin ve yazılım çıktılarının sahiplerini tanımlayan yaşam çevrimi, test planı, iş ve varlık akışını otomatikleştirir [7]. Özelleştirilebilir gösterge panoları

(dashboard) aracılığıyla içeriği uyarlar ve bilgi sunar. Web 2.0 arabirimiyle, farklı yerlerde görev yapan takımlar arasında işbirliği sağlamaktadır. Yazılım geliştirme yaşam çevrimi boyunca oluşturulan yazılım varlıklarına (analiz ve tasarım dokümanları, UML diyagramları, kaynak kodlar, test senaryoları, değişiklik istekleri, vb.) ortak erişimi ve bu varlıkların arasındaki ilişkilerin tutulmasını sağlamaktadır. Zamanla oluşturulan gereksinim tanımlama pratiklerine uyum sağlayabilen "süreç çerçevesi" sunar. Ayrıca testlerin net bir şekilde tanımlanması ve yürütülmesi için zengin metin, yerleşik görüntüler ve yardımlı veri girişi ve doğrulamayı kullanarak manuel olarak test yazma olanağı sunmaktadır. Kullanıcıların, test yürütme zamanlamaları aracılığıyla en iyi ortam kapsamına ulaşmalarına yardımcı olur. Laboratuvar varlıklarını izlemelerine ve zamanlamalarına yardımcı olmak için temel test laboratuvarı yönetim yetenekleri sağlamaktadır. Ortak havuzda, yeniden kullanılmaları için test varlıklarının ve şablonların sürümlü geçmişini saklar. Kapsamlı filtreleme raporlaması ile otomatik veri toplama olanağı sunmaktadır [7].

1.2.4 Microsoft Test Manager 2010

Testçiler ve geliştiriciler arasında bağlamsal işbirliğine olanak tanıyan, komple bir planla, test et, izle iş akışı sunmaktadır. Hata noktasına ulaşmak için gereken manuel adımları otomatikleştirmektedir. Geliştiriciler için, zengin tanımlamalar içeren yüksek kaliteli hata raporları oluşturulmasını sağlamaktadır. Team Foundation Server'la sıkı entegrasyon sayesinde tüm takım rolleri arasında bağlamsal işbirliği sağlar, proje genelinde görünürlüğü büyük ölçüde arttırmaktadır. Bunlarla beraber kullanıcı haberleriyle gereksinimlerini, ilerleme raporlarını ve gerçek zamanlı kalite ölçümlerini eksiksiz izleyebilmektedir [13].

Sanal Laboratuvar Yönetimi, Microsoft'un Ultimate ile MSDN ve Test Professional ile MSDN abonelerine sunduğu Visual Studio Laboratuvar Yönetimi çözümü, mevcut Visual Studio ALM platformunu genişleterek tümleşik Hyper-V tabanlı sanal makine yönetimini olanaklı kılar. Laboratuvar Yönetimi; inşa sürecini optimize etmek, riski azaltmak ve piyasaya sürme sürenizi hızlandırmak için karmaşık inşa – dağıtım - test iş akışlarını otomatikleştirmektedir. Yük ve Web Performansı Testi Microsoft'un Visual Studio Yük ve Web Performansı çözümü, uygulamayı geliştirme yaşam döngüsü boyunca, gerçekçi biçimde modellenmiş yük

simülasyonuyla stres testine tabi tutmanıza imkân vermektedir. Her biri 1.000 sanal kullanıcı barındırabilen sanal kullanıcı paketlerinin esnekliği sayesinde, projelerin değişken ihtiyaçları ne olursa olsun, kritik iş uygulamalarına büyük yükler bindirilebilir [13].

Kodlanmış Kullanıcı Arabirimi Testleri Visual Studio 2010 Premium veya Visual Studio 2010 Ultimate'i kullanarak, bir eylem kaydından kodlanmış UI testi yaratılabilir. Bu test, bir uygulamanın kullanıcı arabiriminin düzgün çalışıp çalışmadığını test edebilir. Kodlanmış kullanıcı arabirimi testi, uygulamanın kullanıcı arabirimi denetimlerinde, çeşitli eylemler gerçekleştirerek doğru denetimlerin, doğru değerlerle görüntülendiğini onaylamaktadır [13].

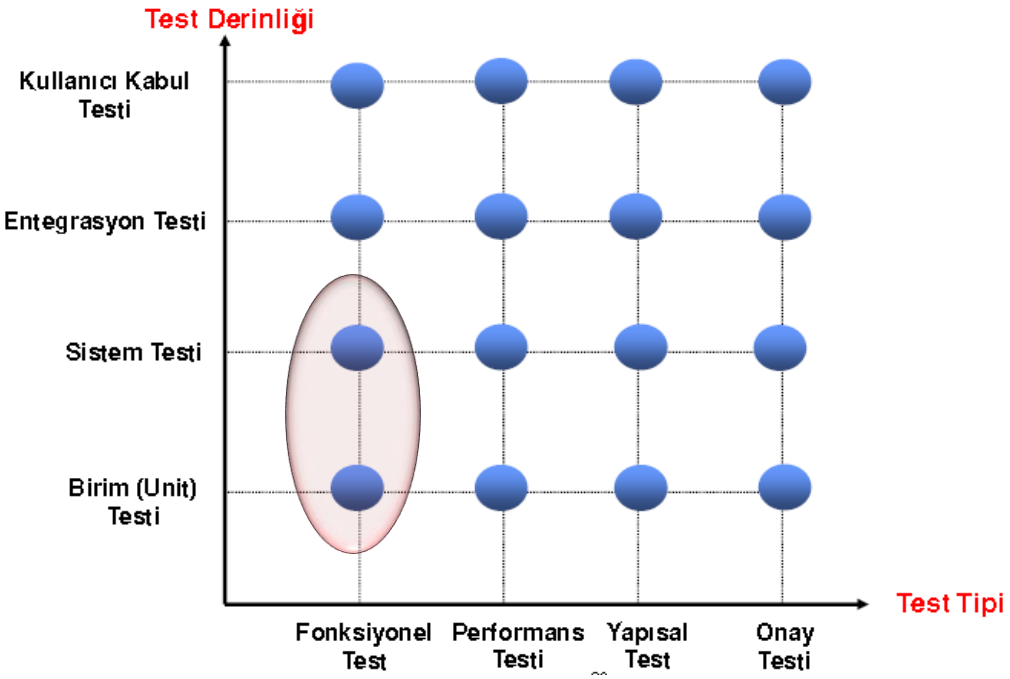
Test Etkisi Analizi, kod değişiklikleri sebebiyle önerilen testlerin listesine erişilmesini ve hangi yapıda, hangi hataların giderildiğinin görülmesini sağlamaktadır [13].

1.3 Geliştirilen Sistemin Genel Yapısı

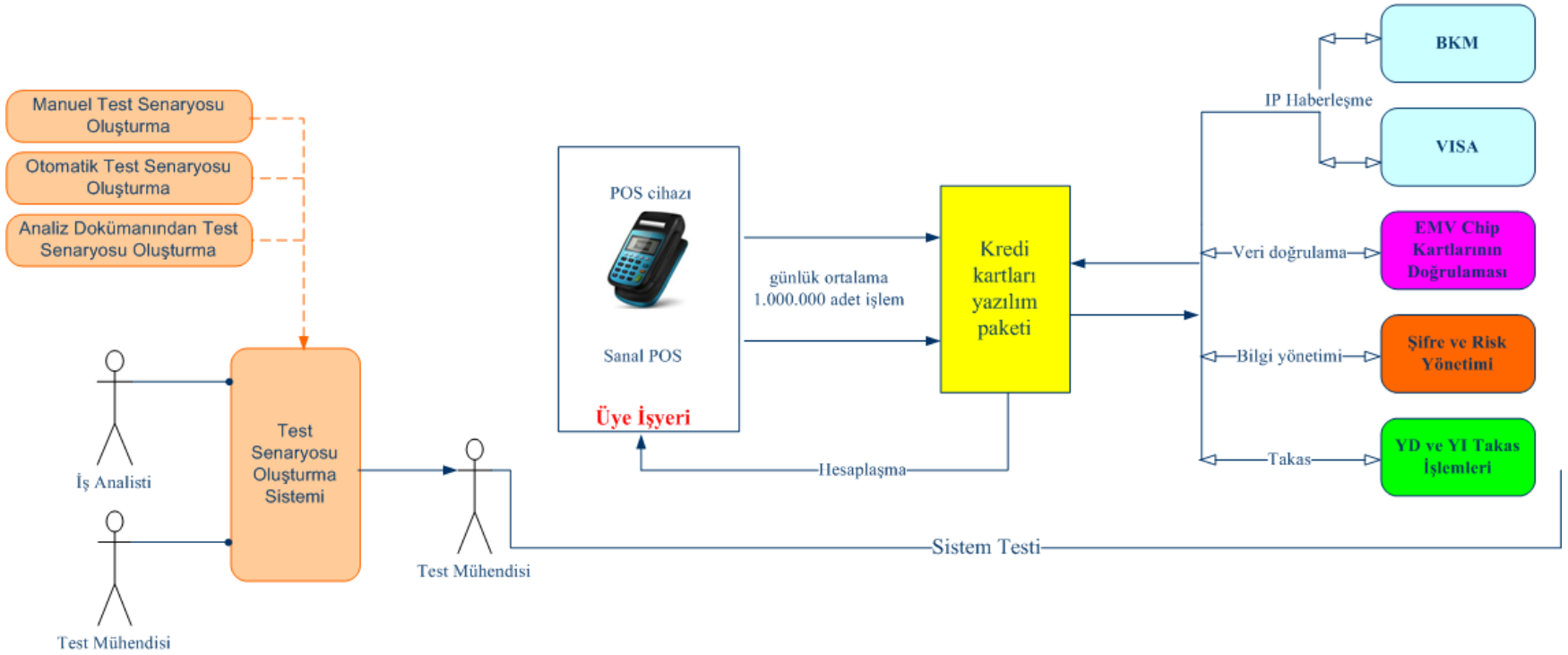
İş analizi ve test ekiplerinin hazırladığı senaryolarının, sistemi uçtan uca kontrol edebilecek derinlikte olması, değişik durumların kontrol edilmesi gerekmektedir. Yoğun çalışma hayatında, test senaryolarının hazırlanması sırasında, detay senaryolar bazen gözden kaçabilmekte ve akla gelmemektedir. Eğer test ekibi de konu hakkında detaylı bilgiye sahip değilse, oluşabilecek her hata, test sürecinde gözden kaçtığı için, maliyette artış meydana getirmekte ve yazılıma olan güven azalmaktadır. Yapılan bu çalışmada, "bankacılık projelerinde başarı oranı daha fazla nasıl yükseltilebilir" sorusunun cevapları, sistem gelişimi sırasında uygulanmıştır.

Geliştirilen sistem, iş analizi ve test ekiplerinin hazırladığı test senaryolarının eksiksiz olmasını sağlamaktadır. Böylece yazılımda çıkabilecek her hata kontrol edilmiş olup, müşteri kullanmaya başladığında yazılımın hatasız çalışması sağlanacaktır. Kredi kartları projelerindeki, testleri yapılmış, sorunsuz çalışan bir modül örnek olarak ele alınmıştır. Modülün, analiz dokümanı kullanılarak, ekranda kontrol edilmesi istenen her parametre belirlenmiştir. Aşağıdaki şekilde sistemin oluşturduğu test senaryolarının derinlik ve tipine göre gösterimi bulunmaktadır.

Geliştirilen sistem, uzman yönlendirme sistemi olup, yazılım testleri için “birim test” ve “sistem testi” seviyesinde, sisteme girişi yapılan parametreler ile manuel, otomatik ve analiz dokümanı ile test senaryoları oluşturarak, ilgili ekiplere yardımcı olmaktadır. Geliştirilen sistemin genel yapısı, Şekil 1.3’te gösterilmektedir.



Şekil 1.2 Geliştirilen sistemin test derinliği ve tipine göre gösterimi



Sekil 1.3 Geliştirilen sistemin yapısı

2. FİZİBİLİTE ÇALIŞMALARI

2.1.1 Ekonomik Fizibilite

Geliştirilen yazılım, iş analistine ve senaryoları hazırlayan test ekiplerine, test adımlarını manuel ve otomatik olarak hazırlayarak yardımcı olacaktır. Herhangi bir ek kurulum gerekmemektedir. Varolan bilgisayarlar üzerine, sistem kurulumu gerçekleştirilebilir. Sistemle birlikte, test senaryolarının hazırlanması için kullanılan kaynak sayısında azalma ve eksik test adımı girişinin engellenmesi sağlanacaktır. Böylece hata bulma oranı artacak, zaman tasarrufuyla müşteri memnuniyeti sağlanmış olacaktır.

2.1.2 Teknoloji ve Kaynak Fizibilitesi

Geliştirilen sistemde kullanılan teknoloji Türkiye'de mevcuttur. Yurtdışı bağımlılığı bulunmamaktadır. Sistemde teknolojik ömür sınırı yoktur. Bilgisayar değişimlerinde kullanıcı programı tekrar kurarak çalışmalarına devam edebilir.

2.1.3 Zaman Fizibilitesi

Sistemin planlanması, sistemin analizi, programın tasarımı ve yazılımı, programın test aşaması için yaklaşık 9 ay gerekmektedir.

Proje başlangıç tarihi : 11.05.2010

Proje bitiş tarihi : 21.01.2011

PLANLAMA

- Problem tanımlama
- Fizibilite raporları
- İş zaman çizelgesi
- Projeye başlama

ANALİZ

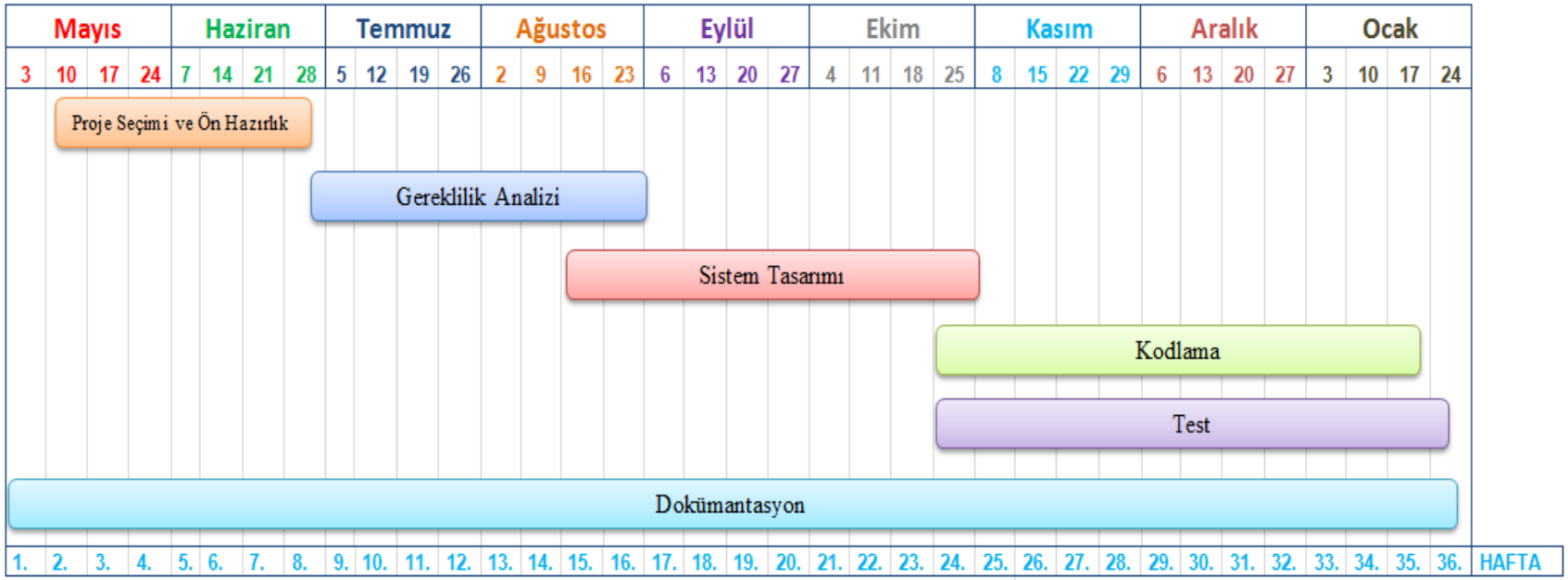
- Bilgilerin toplanması
- İhtiyaçların belirlenmesi
- Gözden geçirilmesi

TASARIM

- Kullanıcı arayüz tasarımı
- Sistem arayüz tasarımı
- Veritabanı tasarımı

UYGULAMA

- Kod yazımı
- Programın test edilmesi



Şekil 2.1 İş - Zaman çizelgesi

3. SİSTEM ANALİZİ

3.1 Yazılım

Geliştirilen sistem için programlama dili olarak, Microsoft'un gelistirmis olduğu yeni nesil bir dil olan C# seçilmiş, Microsoft Visual Studio 2010 geliştirme aracı kullanılmıştır.

Sistemin veri tabanı ile iliskisinin SQL (Structured Query Language) sorgulama diliyle gerçekleştirilmiştir. Veri tabanı için Microsoft SQL Server 2008 sunucusu seçilmiştir.

Sistem içinde yer alan, Excel'e tablo içeriği gönderme özelliği için, The Office 2010 Primary Interop Assemblies Redistributable paket kurulumu yapılacaktır.

Ayrıca programın çalışması için gerekli olan Microsoft .NET Framework Version 2.0 paket kurulumu yapılacaktır.

3.2 Veri Tabanı

Geliştirilen sistemin veri tabanı sunucusu olarak MSSQL kullanılmıştır. Sistem sunucu üzerinden bilgileri okuyacak, güncelleme ve kaydetme işlemleri yapacaktır.

3.3 Geliştirilen Sistem ve Alternatif Çözümler

Uzman sistemler ile geleneksel sistemler arasındaki en büyük farklardan birisi karşılaştırma yeteneğidir. Geleneksel sistemler, uzman sistemlerin aksine karşılaştırma gerektiren konularda zayıf kalmaktadır [17]. Uzman sistemler, problem çözümünde hiyerarşik bir yaklaşım izlemektedir. Nümerik veri ve algoritmalardan ziyade gerçek kurallar ve ilişkilerden oluşmakta, problem çözerken ve tanımlarken kullanıcıya danışmaktadır. Uzman sistemlerde, danışma yapısı ve şekli elde bulunan bilgiye, probleme ve soru şekline göre değişim göstermektedir. İstenildiğinde, problem çözümü sonucuna ulaşmadan ara sonuç verebilmekte, belli bir soruyu neden

sorduklarını veya belli bir sonuca nasıl ulaştıklarını açıklayabilmektedir. Kesin veya tam olmayan bilgilerle başedilebilmektedir (örneğin, kullanıcı sisteminin sorularını "belki" veya "bilmiyorum" diye cevaplamış olabilir). Gerçek bir problemi çözmek için, bilgiler tam olmazsa bile yaklaşık bir sonuç verebilmeleri, çözülen problemin sonuçlarını doğal bir dille açıklamaları, bir kere kurulduktan sonra kullanıcı tarafından, programcıya ihtiyaç duyulmadan, kolayca geliştirilip değiştirilebilmeleri, problemleri çözerken, çözülen problemin verilerini daha sonraki çözümlerde kullanmak amacıyla kendi bilgilerine ilave ederek kendisini geliştirmesi uzman sistemlerin önemli özelliklerindendir [17].

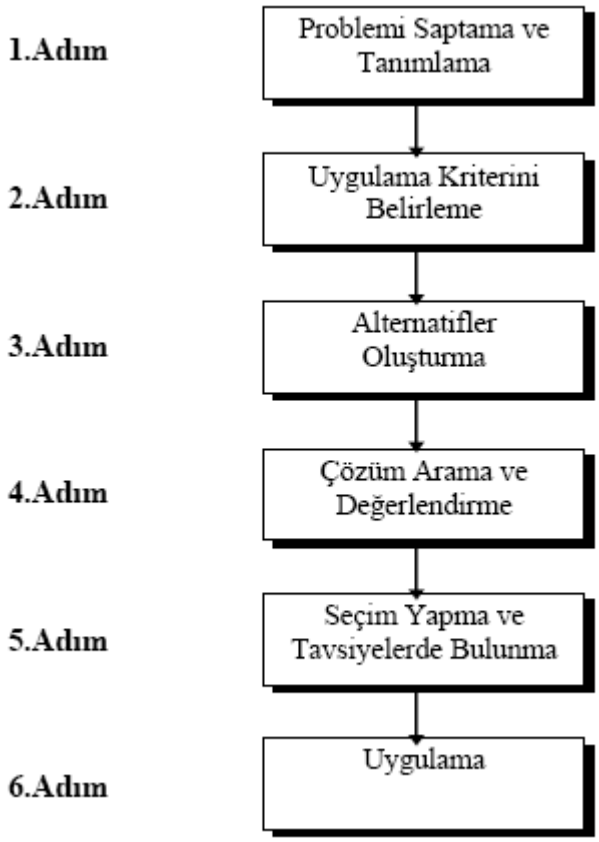
Uzman sistemler teknolojisi yapay zekaya bağlı olarak ortaya çıkmıştır. Yapay zeka ise, bilgisayar bilimlerinin; problem çözme, optik algılama ve doğal dilleri anlama gibi insani yeteneklere sahip programların tasarımı ve işletmesi ile ilgili dalıdır. Geliştirilen bankacılık yazılımında test senaryosu üretebilen uzman sistem, uygulamalı yapay zeka olarak kabul edilmektedir.

Problem çözme, genellikle altı temel adım iş akışı içinde gözlenebilir: problemi saptama ve tanımlama, bir çözüm bulmak için kriter belirleme, alternatifler oluşturma, çözüm arama ve değerlendirme, seçim yapma, tavsiyelerde bulunma ve uygulama [20].

Şekil 3.1'de iş akışı lineermiş gibi gösterilmesine rağmen nadiren lineerdir. Gerçek hayatta bu adımlardan bazıları birleştirilebilir, bazı adımlar temel adım olabilir ya da başlangıç adımlarında revizyonlar yapılabilir. Her bir adımın kısa açıklaması aşağıda verilmektedir.

Problemi Belirleme ve Tanımlama: Bir problem (ya da fırsat) ilk olarak farkedilmelidir. Problemin (ya da fırsatın) büyüklüğü ve önemi belirlenir ve tanımlanır [20].

Kriteri Belirleme: Bir problemin çözümü, olası alternatifleri karşılaştırmak için kullanılan kritere bağlıdır. Örneğin iyi bir yatırım arama, güvenlik, likidite ve geri çevirme oranı gibi kriterlere bağlıdır. Bu adımda kriterleri ve biribirine göre bağıl önemleri belirlenmektedir [20].



Şekil 3.1 Problem çözme iş akışı

Alternatifler Oluşturma: Tanıma göre bir karar durumu olabilmesi için iki ya da daha fazla hareket şansı olmalıdır. Potansiyel çözümler yaratma, yaratıcılık ve zeka gerektirir [20].

Çözüm Arama ve Değerlendirme: Bu adımda önceden belirlenmiş kriterler ışığında çözüm seçenekleri incelenir. En iyi ya da yeterince iyi çözümleri bulmaya çalıştığımız için temel olarak bir arama prosesidir. Bu adımda birkaç tane arama, değerlendirme ve sebep bulma metodolojileri kullanılabilir [20].

Seçim Yapma ve Tavsiyelerde Bulunma: Aramanın sonucu, bir probleme çare olarak tavsiye etmek için bir çözüm seçmedir.

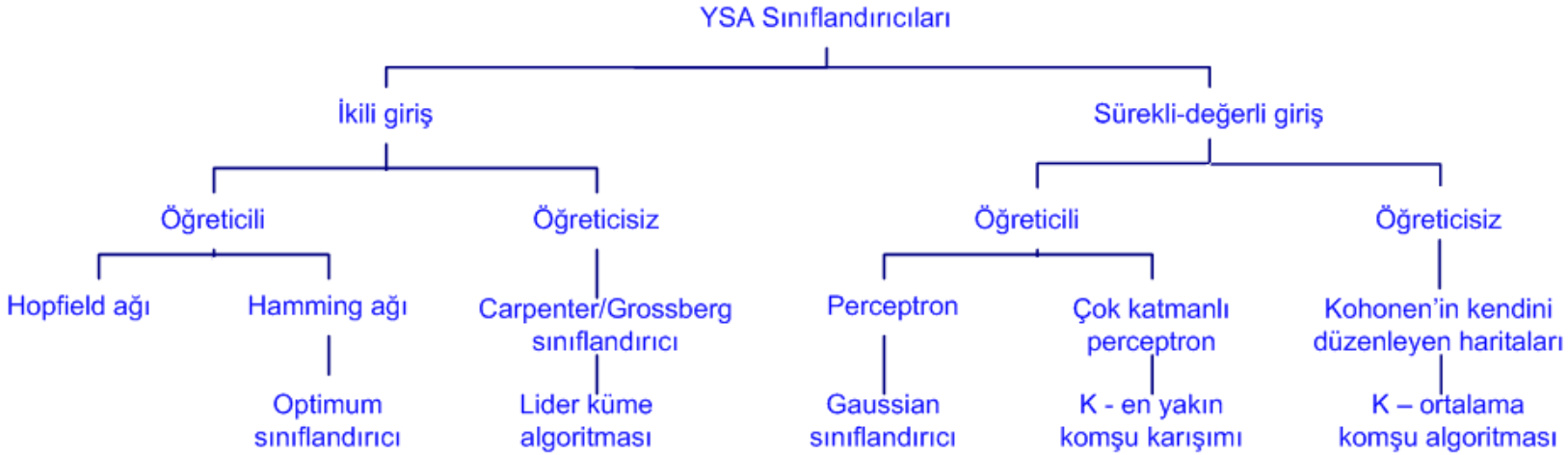
Uygulama: Problemi çözmek için tavsiye edilen çözüm başarılı olarak gerçekleştirilmelidir. Aslında bu iş akışı daha karmaşıktır. Çünkü her adımda herbiri benzer prosesi takip eden birkaç ara kararlar olabilir [20].

Uygulamalı yapay zeka teknolojileri bu altı adımın hepsini desteklemek için kullanılabilir. Fakat yapay zeka hareketlerinin çoğu 4. ve 5. adımlarda gerçekleşir. Özellikle uzman sistemler, verildiği farzedilen alternatiflerden çözüm bulmak için kullanılır. Yapay zekanın rolü temel olarak aramayı ve değerlendirmeyi, bazı sonuç çıkarma yeteneklerini kullanarak yönetmektir. Bugün yapay zekanın sınırlı rolüne rağmen belirli bir zaman sonra teknlojilerin, iş akışı adımlarında daha fazla rol oynayacağı ümit edilmektedir. Yapay zekanın etkili olarak problem çözme iş akışının sadece iki adımını kullanmasına rağmen yapay zeka, problem çözme ve karar verme olarak sınıflandırılmayan diğer birçok görevde kullanılmaktadır. Uygulamalı yapay zeka teknolojileri, problem çözme iş akışının arama ve değerlendirme adımlarıyla birleştirilir [20].

Problemlere uygun çözümleri bulmak için birçok arama yöntem ve stratejiler kullanılmaktadır. Bu yöntemler optimizasyon, blind arama ve heristik kullanımı olarak üç kategoriye göre sınıflandırılabilir. Blind arama ve heristik sayısal ya da nitel (sembolik) analiz gerektirirken optimizasyon sayısal ve nicel analiz gerektirir. Optimizasyon, belirli bir durumu modelleyen matematik formülleri kullanarak mümkün olan en iyi çözümü bulmaya çalışır. Problem alanı kurallara uygun bir biçimde yapılandırılmalıdır [20].

Yapılan bu çalışmada, uygulamanın geliştirildiği dil olarak C# ve ortam olarak Microsoft Visual Studio seçilmiştir. Çeşitli algoritmalar kullanılarak test senaryosu oluşturma işlemi gerçekleştirilmiştir. Geliştirilen sistem, uzman yönlendirme sistemi olup, yazılım testleri için “birim test” ve “sistem testi” seviyesinde, test senaryolarını manuel ve otomatik olarak oluşturarak, ilgili ekiplere yardımcı olmaktadır. Alternatif bir çözüm olarak, yapay sinir ağı (YSA) sınıflandırıcıları kullanılarak, sisteme girişi yapılan parametrelerle karar verilmesi ve test senaryosu üretilmesi sağlanabilir. Fakat bu çözüm, sistem gereksinimini tam olarak sağlamayacaktır. İhtiyaç duyulan, varolan yapıda eksikliği hissedilen, test aşamasında baştan sona sistemin kontrol edilmesini sağlayacak özellikleri, uzman yönlendirme sistemi karşılamaktadır. YSA, paralel dağılmış bir bilgi işleme sistemidir. Bu sistem genellikle tek yönlü işaret kanalları (bağlantılar) ile birbirine bağlanan işlem elemanlarından oluşur. Çıkış işareti bir tane olup isteğe göre çoğaltılabilir. YSA yaklaşımının temel düşüncesiyle, insan beyninin fonksiyonları

arasında benzerlik vardır. Bu yüzden YSA sistemine “insan beyninin modeli” denilebilir. YSA çevre şartlarına göre davranışlarını şekilleyebilir. Girişler ve istenen çıkışların sisteme verilmesi ile kendisini farklı cevaplar verebilecek şekilde ayarlayabilir. Ancak son derece karmaşık bir iç yapısı vardır.



Şekil 3.2 YSA sınıflandırıcıları

4. SİSTEM TASARIMI

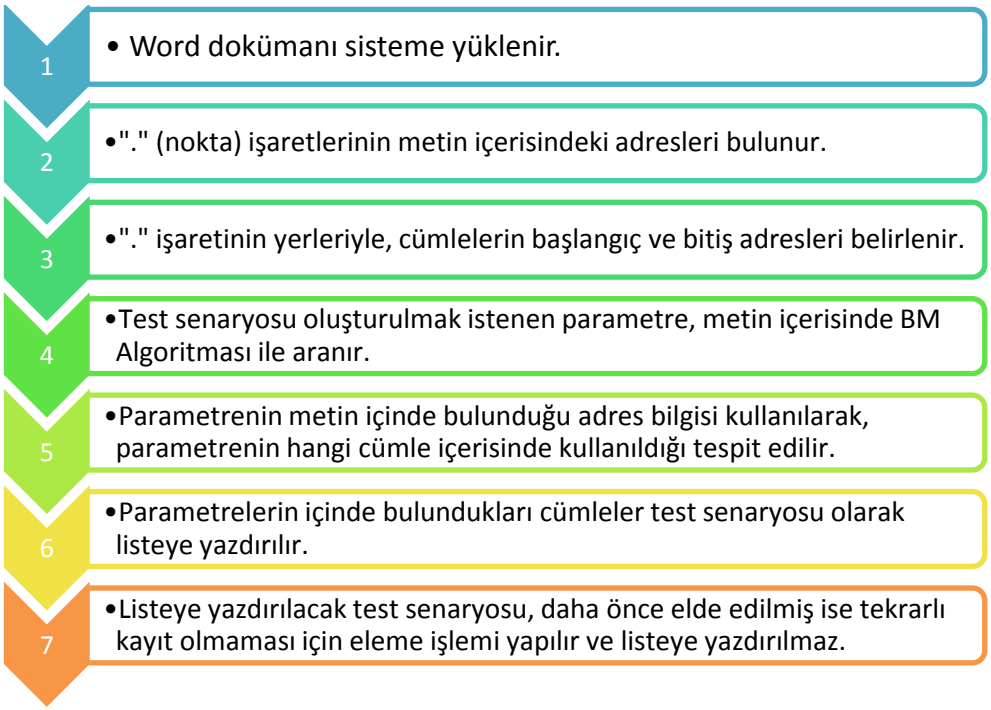
Sistem geliştirmede kullanılan algoritmalar, veri tabanı ve metotlar, bu bölümde detaylı olarak anlatılmaktadır.

4.1 Boyer Moore Algoritması

Bir metin veya hedef dizgi (string) içerisinde bir başka dizginin aranması sırasında kullanılan algoritmalardan birisidir. KMP (Knuth Morris Prat) algoritması ile birlikte en çok kullanılan arama algoritmalarındandır [11]. Boyer-Moore (BM) Algoritması, Türkçe metinler üzerinde, arama işlemini daha kısa sürede gerçekleştirdiği için geliştirilen uygulamada kullanımı tercih edilmiştir.

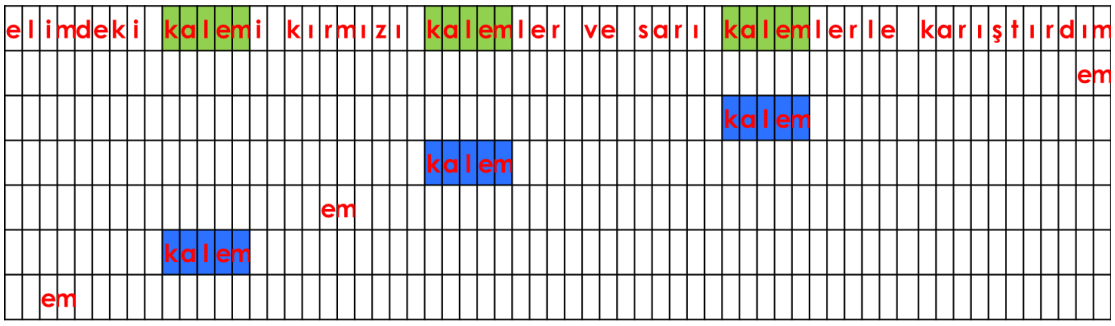
BM Algoritması dizgi karakterlerinin en sağındaki karakterden başlayarak sağdan sola doğru arama işlemini gerçekleştirir. Uyuşmama durumunda sol tarafa kaydırma işlemi yapılarak karşılaştırma yapılır.

Sistemde, analiz dokümanına göre test adımlarını oluşturmak için kontrol edilmek istenen parametrelerin aranması BM algoriması ile sağlanmıştır. Metin içerisindeki "." (nokta) işaretleri ise klasik arama yöntemiyle elde edilmiştir. Modülde, okunmak istenen Word dokümanı seçilerek, metin sisteme yüklenir. Metin içerisindeki tüm "." işaretlerinin yerleri bulunarak, dizi içerisinde cümle başlangıç ve bitiş adresleri olarak tutulmaktadır. Cümlelerin yerlerini belirten dizi, metin içerisinde aranan her parametrenin test senaryosu olarak oluşturulmasında kullanılmaktadır. Tek parametre ya da listede bulunan parametrelerden toplu arama yapmak için, BM algoritması çalıştırılır. Algoritma çıktısında aranan parametrelerin bulundukları yer belirlenmektedir. Eldedilen parametre yer bilgisi kullanılarak hangi cümle içerisinde yer aldığı bulunabilmektedir. Parametrelerin içerisinde geçtiği cümlelerin belirlenmesiyle, test senaryosu olarak cümleler listeye yazdırılır. Listeye yazdırılacak test senaryosu, daha önce elde edilmiş ise tekrarlı kayıt olmaması için eleme işlemi yapılır ve listeye yazdırılmaz.

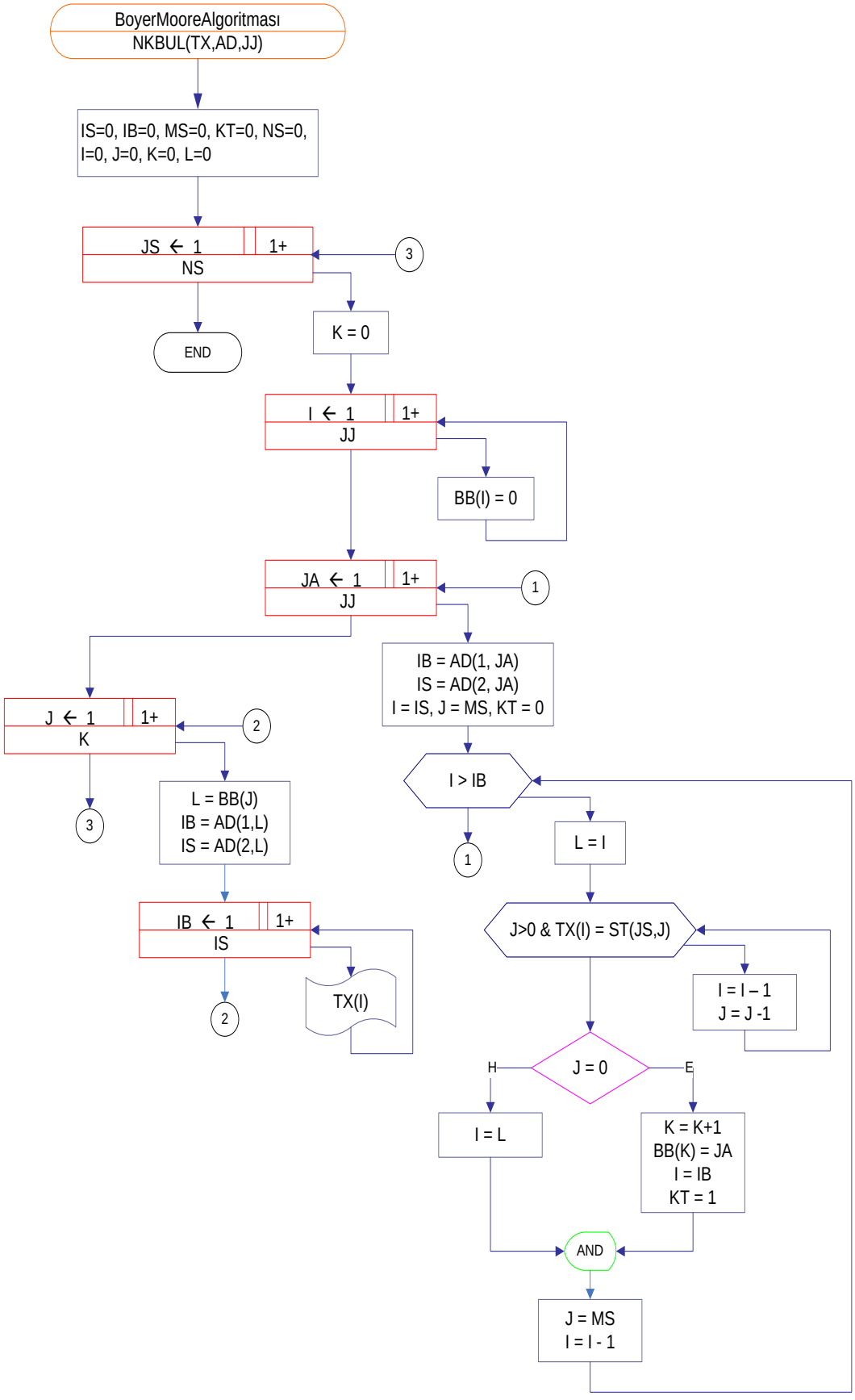


Şekil 4.1 BM Algoritmasının içinde bulunduğu iş akışı

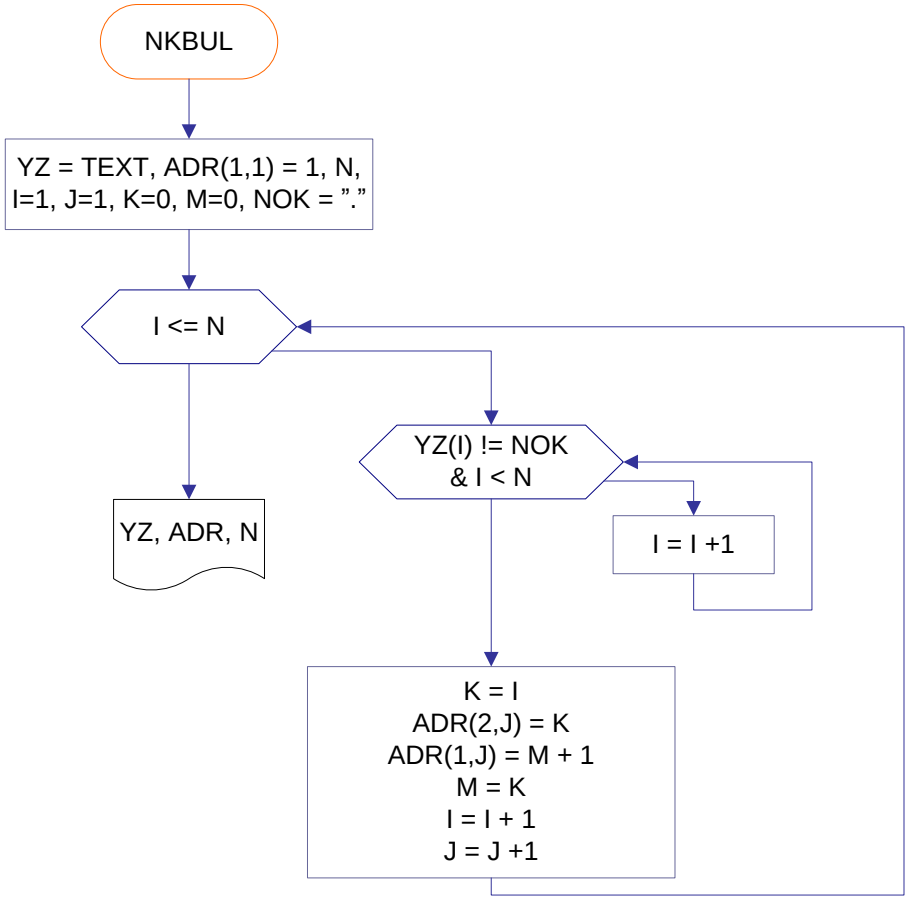
Aşağıda verilen örnek cümlede "kalem" kelimesi aranmaktadır. BM algoritması, aranan kelimenin son harfi olan "m" harfini, cümlenin sonundan başına doğru aramaya başlamaktadır. "m" harfinin cümlede bulunduğu ilk yerde aranan kelimenin diğer harfleri karşılaştırılmaya başlanır. Örnekteki ilk satırda "e" harfi, "ı" harfi ile karşılaştırıldığında eşleşme olmadığı için "m" harfi cümle içinde aranmaya devam eder. İkinci satırda görüldüğü gibi, ilk "kalem" kelimesi bulunmuş olunur. Cümle başına gelene kadar arama işlemi devam ederek bulunan kelimelerin adresleri belirlenmektedir.



Sistemde kullanılan BM algoritması ve alt fonksiyonunun akış şemaları aşağıdaki gibidir.



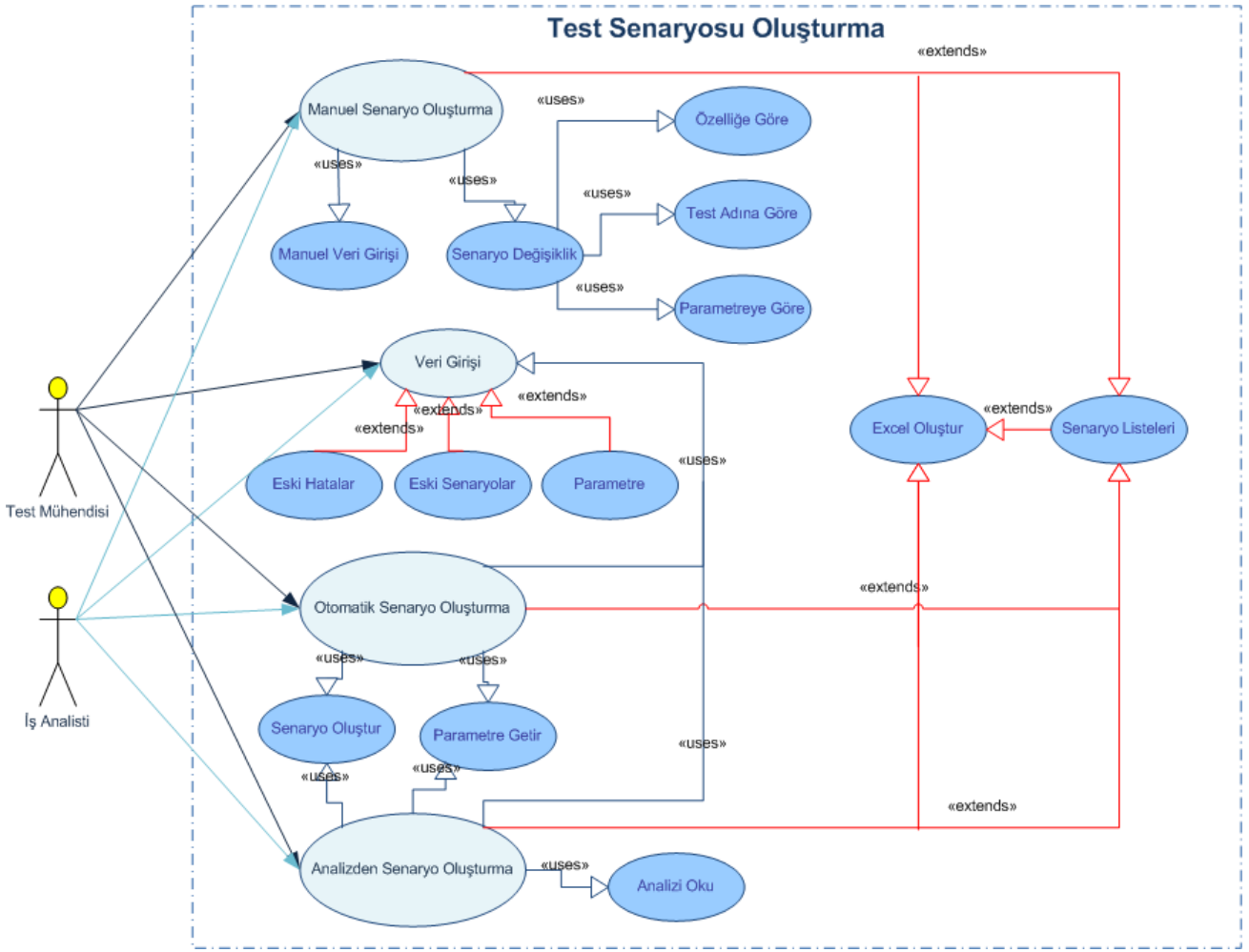
Şekil 4.2 Boyer Moore Algoritması



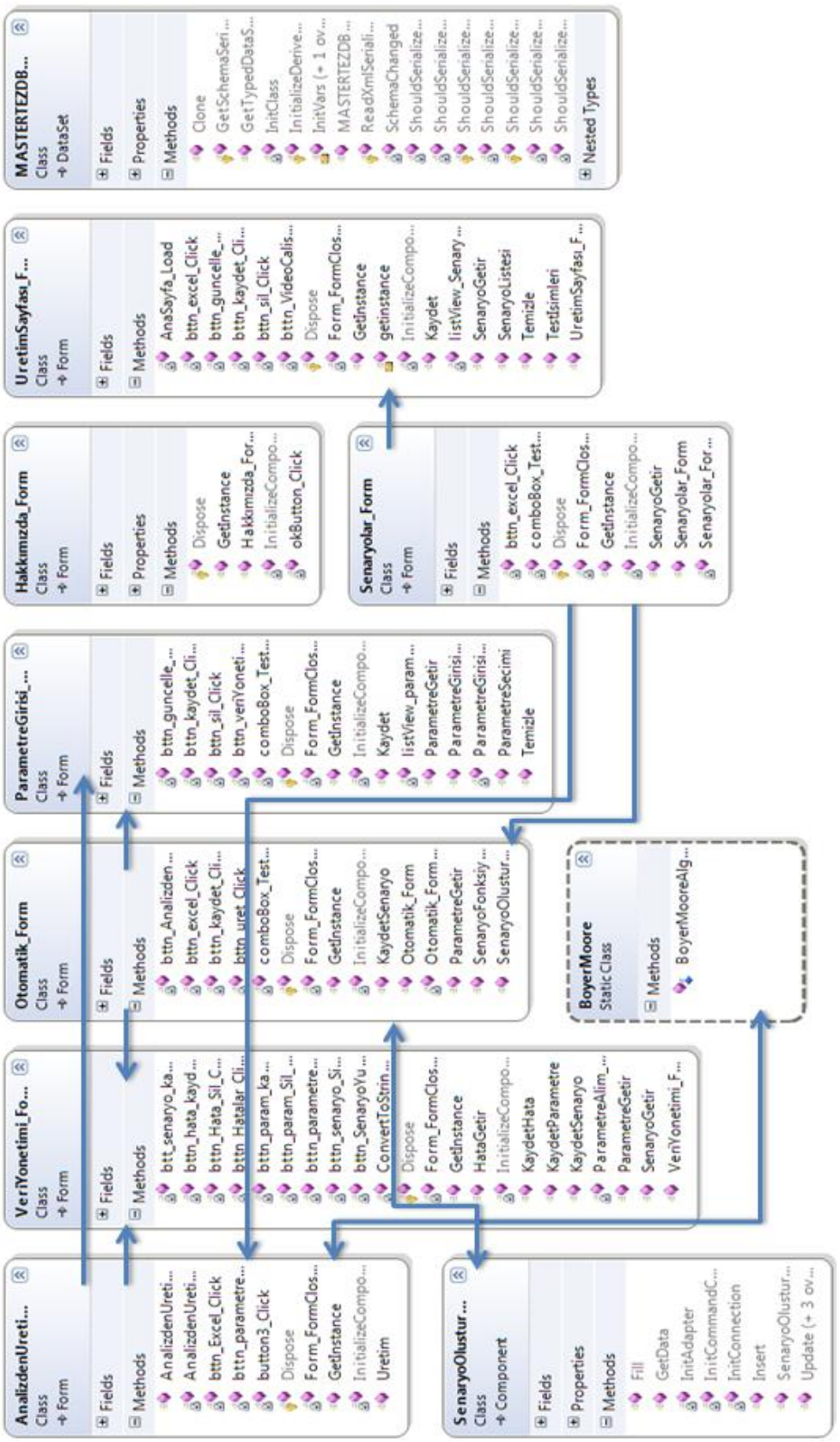
Şekil 4.3 Nokta adreslerini bulma fonksiyonu

4.2 Geliştirilen Sistemin Yapısı

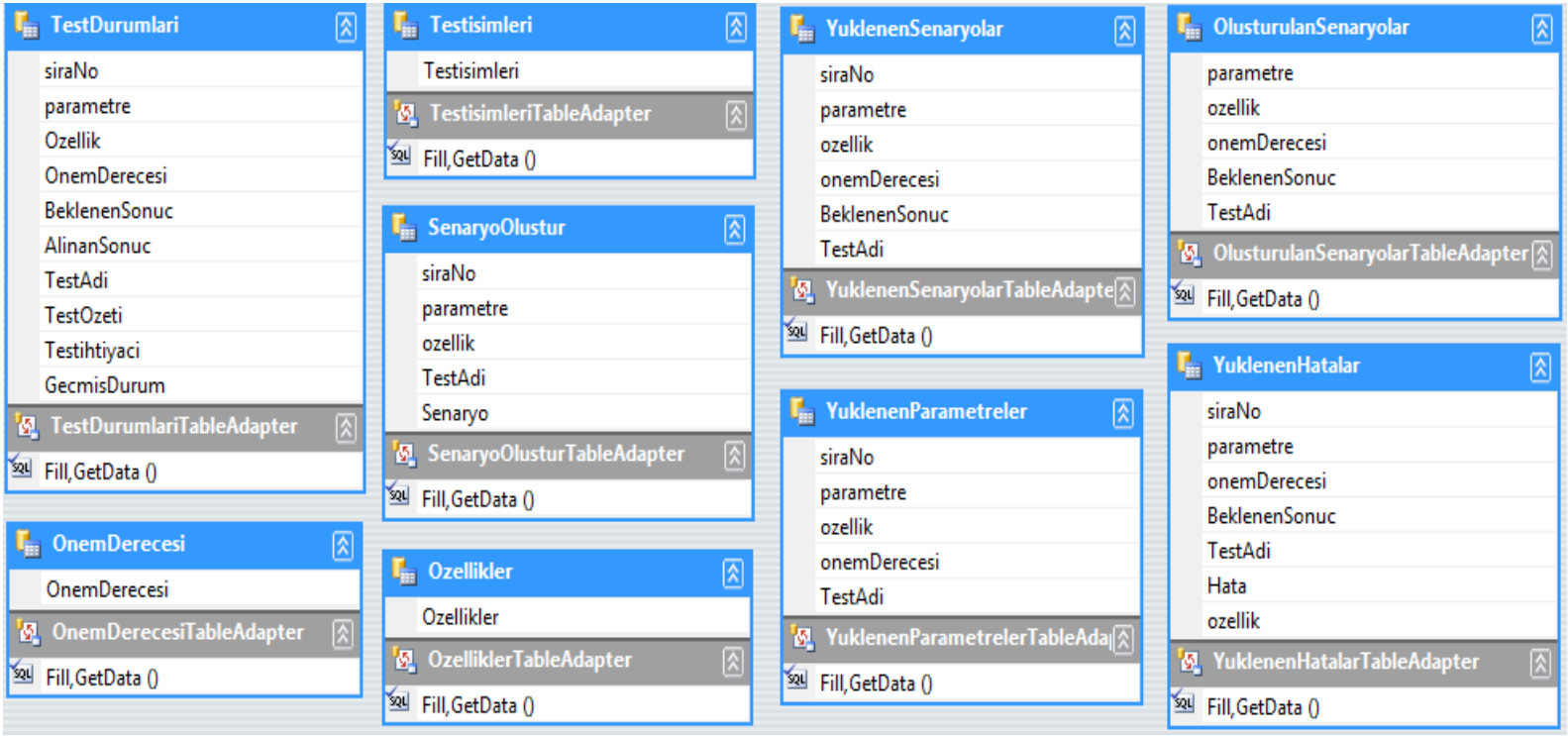
Sistem altı modül ve bir arayüzden oluşmaktadır. Altı modülden üçü test senaryolarını oluşturmakta, iki modül test senaryolarının oluşması için veri girişinin yapılmasında yardımcıdır. Altıncı modül ise test senaryolarını gösterme amaçlıdır. Arayüzde de sistemle ilgili bilgiler verilmektedir. Sistem genel olarak, girişi yapılan parametrelerle ilgili test senaryolarının oluşturulmasını sağlamaktadır. Test senaryolarının oluşturulması için varolan senaryo veritabanı yada analiz dokümanı kullanılmaktadır. Ayrıca manuel olarak, test senaryolarını kullanıcı da oluşturabilmektedir. Aşağıdaki şekillerde sistemin, UML diyagramı, kullanılan metotları ve veri tabanı gösterilmiştir.



Şekil 4.4 Sistemin UML diyagramı



Şekil 4.5 E-R diyagramı



Şekil 4.6 Veri tabanı tabloları

4.3 Test Edilecek Yazılımın Gereklilik Analizi

Müşterinin istekleri doğrultusunda, test edilecek ekran için hazırlanan analiz maddelerine aşağıda yer verilmiştir. Geliştirilen sisteme, parametre girişleri analiz sonuçlarına göre yapılacaktır.

İş analistleri tarafından hazırlanan gereksinim analiz dokümanlarında, tasarlanacak yazılımda müşterinin olmasını istediği özelliklere yer verilmektedir. Geliştirilen uzman yönlendirme sisteminin uygulaması için kullanılan analiz dokümanında yer alan özelliklerin başlıkları aşağıda bulunmaktadır.

- Sorgulama tipi
- Sorgulama kriterleri
- Veri giriş alanları
- Fonksiyonel butonlar
- Aktif ve inaktif alanlar
- Ekran kontrolleri
- Liste alanları

Tablo 4.1 Test edilecek görsel öğeler

Öğe Adı	Öğe Tipi	Durum	Zorunlu Alan
Sorgulama Tipi	Seçim düğmesi	Aktif	Evet
İşyeri No	Seçim düğmesi	Aktif	Evet
Bölüm No	Seçim düğmesi	Aktif	Hayır
Terminal No	Seçim Listesi	Aktif	Hayır
Source	Seçim Listesi	Aktif	Hayır
Brand	Seçim Listesi	Aktif	Hayır
Banka – Kredi	Seçim Listesi	Aktif	Hayır
Banka Kodu	Seçim düğmesi	Aktif	Hayır
Kampanya Kodu	Seçim düğmesi	Aktif	Hayır
Kampanya Tipi	Seçim Listesi	Aktif	Hayır
İşlem Türü	Seçim düğmesi	Aktif	Hayır
Para Birimi	Seçim düğmesi	Aktif	Hayır
İşlem Tarihi	Seçim Listesi	Aktif	Evet
Taksit Tarihi	Seçim Listesi	Aktif	Evet
Günsonu Tarihi	Seçim Listesi	Aktif	Evet

Beklenen Hesaba Geçiş Tarihi	Seçim Listesi	Aktif	Evet
Erken Ödeme Tarihi	Seçim Listesi	Inaktif	Evet
Taksit Sayısı	Numerik Alan	Aktif	Hayır
Erken Ödeme Talep No	Seçim Listesi	Inaktif	Hayır

4.3.1 İşyeri uygulamalarına ilişkin genel kavramlar

İşyeri: İşyerinin aynı adreste faaliyette bulunan POS (point of sale) cihazlarının izlenmesi amacı ile kullanılan işyeri numarasıdır. POS cihazı haricindeki farklı satış yöntemlerinin (sanal POS, posta/telefon ile satış, vs.) izlenmesi amacı ile farklı işyeri numaraları açılmaktadır.

Merkez: Aynı ticari unvana sahip ve aynı faaliyet konusu ile iştigal eden işyerlerinin izlenmesini sağlayan altyapıdır.

Ana merkez: Bayilik sistemi ile çalışan üye işyerlerinin izlenebilmesi amacı ile kullanılan işyeri altyapısıdır.

Bölüm: Aynı işyeri numarası altında farklı mâli koşullar veya farklı cari hesapların tanımlanabilmesini sağlayan altyapıdır.

YI/YD Kart Koşulu: İşyerinde yurtiçi ve yurtdışı bankaların kartları ile gerçekleştirilen işlemlerde her kart tipi için geçerli olacak mali koşul bilgileridir.

Banka Kart Koşulu: İşyerinde, anlaşmalı bankanın kartları ile gerçekleştirilen işlemlerde geçerli olacak mali koşul bilgileridir.

Tablo 4.2 Test edilecek liste alanları tablosu

Sıra	**Alan adı**	**Sıra**	**Alan adı**
1	Merkez no	31	Para birimi
2	Merkez tabela adı	32	Para birimi açıklaması
3	İşyeri no	33	İşlem tutarı
4	Tabela adı	34	Taksit tarihi
5	Vkn/tckn	35	Taksit sayısı
6	Cari hesap no	36	İşyeri taksit sayısı
7	Bölüm no	37	İşyeri taksit no

8	Anamerkez no	38	Taksit tutarı
9	Anamerkez tabela adı	39	Beklenen hesaba geçiş tarihi
10	Şube kodu	40	Merkez hizmet komisyon oranı
11	Terminal no	41	Merkez hizmet komisyon tutarı
12	İşlem kaynağı	42	Anamerkez hizmet komisyon oranı
13	Pos Entry Mode	43	Anamerkez hizmet komisyon tutarı
14	İşlem tarihi	44	Banka puan oranı
15	İşlem zamanı	45	Banka puan tutarı
16	Otorizasyon kodu	46	Net tutar
17	Source	47	İşyeri puan oranı
18	Brand	48	İşyeri puan tutarı
19	Banka - kredi	49	Merkez puan oranı
20	Kart numarası	50	Merkez puan tutarı
21	Banka adı	51	Anamerkez puan oranı
22	Kampanya kodu	52	Anamerkez puan tutarı
23	Mali koşul kampanya adı	53	İşyeri Komisyon/Prim Oranı
24	Kampanya tipi	54	İşyeri Komisyon/Prim Tutarı
25	İşlem Türü	55	İşyeri Hizmet Komisyon Oranı
26	İşlem türü açıklaması	56	İşyeri Hizmet Komisyon Tutarı
27	İşlem Tipi	57	İskonto Oranı
28	İşlem kodu	58	İskonto Tutarı
29	İşlem alt kodu	59	Ödeme Gün Bilgisi
30	Geri Ödeme Tipi		

Teorik olarak analizi yapılan ve geliştirilen sistemin uygulaması, örnek olarak alınan kredi kartları projelerindeki, testleri yapılmış, sorunsuz çalışan bir modülün test senaryoları oluşturularak gerçekleştirilmiştir. Modülün, analiz dokümanı kullanılarak, ekranda kontrol edilmesi istenen her parametre belirlenmiş, bu parametreler ile geliştirilen sistemde manuel, otomatik ve analiz dokümanından test senaryoları oluşturulmuştur. Elde edilen test senaryolarının, testleri yapılacak ekranı baştan sona kontrol edebilecek derinlikte olduğu görülmüştür. Test senaryolarının başarılı şekilde geliştirilen sistem tarafından oluşturulması, iş analizi ve test ekiplerinin, sistemden beklentilerini karşılamaktadır.

4.4 Modül İnceleme

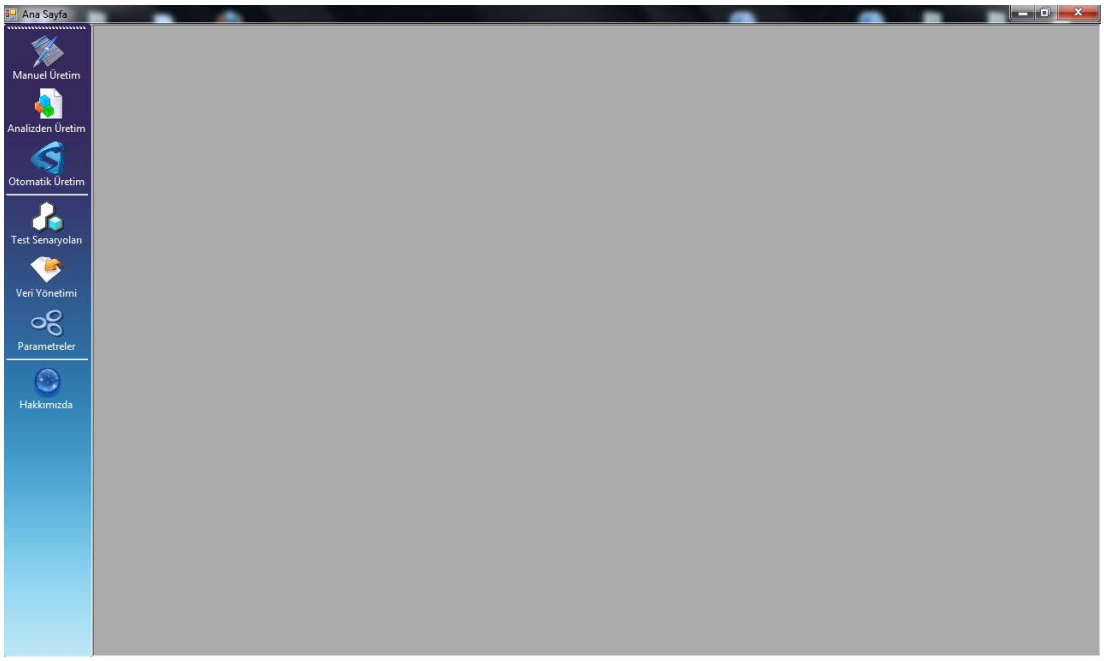
Geliştirilen sistemin modüllerinin kullanımı detaylı olarak aşağıdaki başlıklarda anlatılmaktadır.

4.4.1 Giriş Ekranı

Sisteme girişte açılan ve diğer ekranlara erişim sağlanan bir ekrandır. Sistem üretim modüllerinden ve sorgulama, veri yönetimi modüllerinden oluşmaktadır.

- Üretim Modülleri
  - Manuel Üretim
  - Otomatik Üretim
  - Analiz Dokümanından Üretim
- Sorgulama ve Veri Yönetimi Modülleri
  - Test Senaryoları
  - Veri Yönetimi
  - Parametreler

Bu ekran üzerinden, sol tarafta bulunan butonlar kullanılarak; manuel üretim, analizden üretim, otomatik üretim, test senaryoları, veri yönetimi, parametreler ve hakkımızda ekranlarına erişim sağlanabilmektedir.



Şekil 4.7 Giriş ekranı

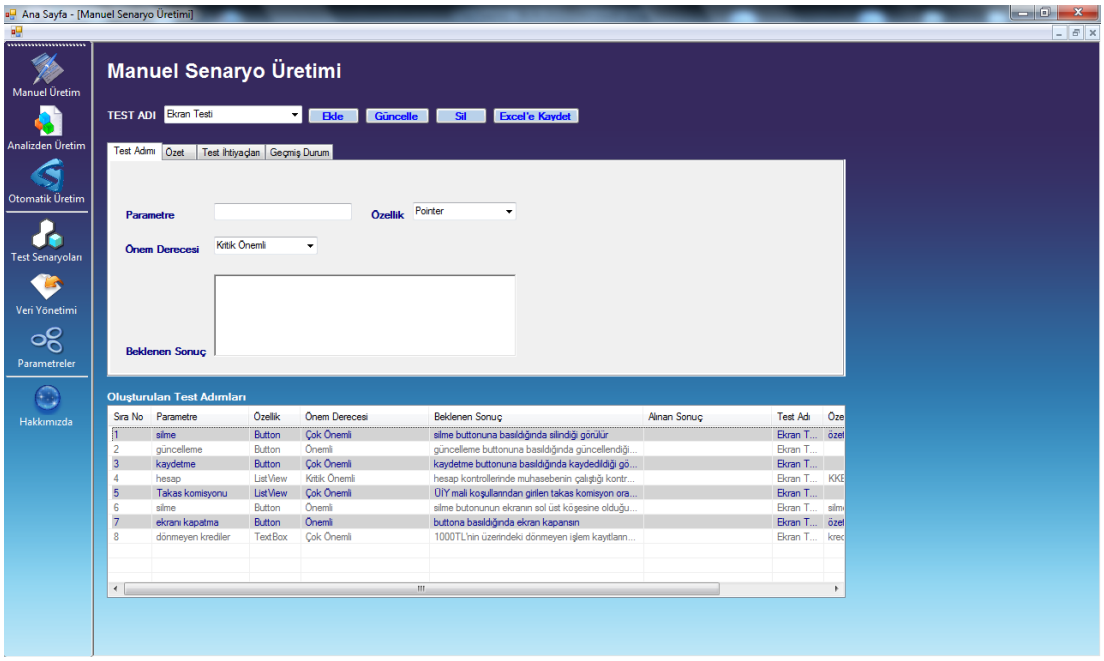
Tablo 4.3 Giriş ekranı görsel öğeleri

Öğe Adı	**Öğe Tipi**	**Durum**	**Zorunlu Alan**
Manuel Üretim	Button	Aktif	Hayır
Analizden Üretim	Button	Aktif	Hayır
Otomatik Üretim	Button	Aktif	Hayır
Test Senaryoları	Button	Aktif	Hayır
Veri Yönetimi	Button	Aktif	Hayır
Parametreler	Button	Aktif	Hayır
Hakkımızda	Button	Aktif	Hayır

4.4.2 Manuel Senaryo Üretim Ekranı

Test edilecek yazılım bilgilerinin manuel olarak girişinin yapıldığı ekrandır. Ekrandaki test adımı sekmesinde; parametreler ve bu parametrelerin özellikleri, önem derecesi ve beklenen sonuç bilgi girişleri yapılır. Özet sekmesinde; test edilecek ekranla ilgili genel bir bilgi verilir. Test ihtiyaçları sekmesinde; testin yapılabilmesi için ihtiyaç duyulan bilgilerin girişi yapılmaktadır. Geçmiş durum sekmesinde ise; test edilecek ekranın daha önce testi yapılmışsa dikkat edilmesi

gereken durumların girişi yapılabilir. Girişi yapılan bilgiler; sisteme kaydedilir, güncellenebilir, silinebilir ve Excel dokümanı olarak alınabilmektedir.



Şekil 4.8 Manuel senaryo üretim ekranı

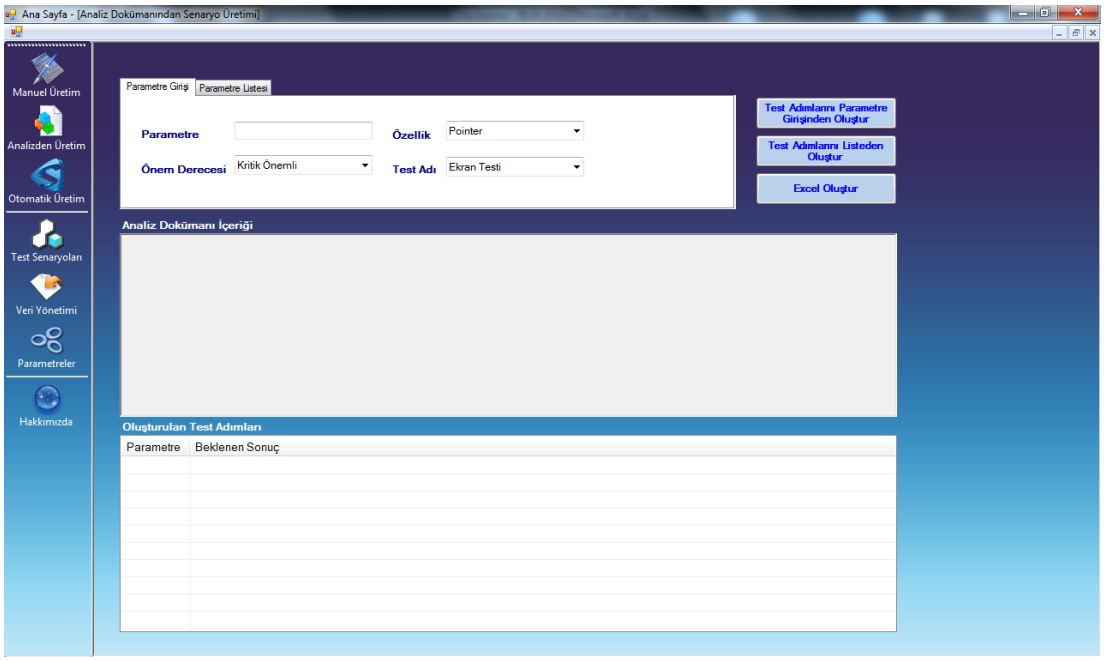
Tablo 4.4 Manuel senaryo üretim ekranı görsel öğeleri

Öğe Adı	Öğe Tipi	Durum	Zorunlu Alan
Test Adı	Seçim Listesi	Aktif	Evet
Ekle	Buton	Aktif	Hayır
Güncelle	Buton	Aktif	Hayır
Sil	Buton	Aktif	Hayır
Excel'e Kaydet	Buton	Aktif	Hayır
Test Adımı Sekmesi	Sekme	Aktif	Hayır
Özet Sekmesi	Sekme	Aktif	Hayır
Test İhtiyaçları Sekmesi	Sekme	Aktif	Hayır
Geçmiş Durum Sekmesi	Sekme	Aktif	Hayır
Parametre	Metin Alanı	Aktif	Evet
Özellik	Seçim Listesi	Aktif	Evet
Önem Derecesi	Seçim Listesi	Aktif	Evet
Beklenen Sonuç	Metin Alanı	Aktif	Evet
Özet	Metin Alanı	Aktif	Hayır
Test İhtiyaçları	Metin Alanı	Aktif	Hayır
Geçmiş Durum	Metin Alanı	Aktif	Hayır
Oluşturulan Test Adımları	Liste	Aktif	Hayır

4.4.3 Analizden Üretim Ekranı

Bu ekranda, test adımları iki farklı yöntemle analiz dokümanından oluşturulmaktadır. İlk yöntemde, testi yapılacak yazılımın kontrol edilmesi istenen parametreleri, tek tek manuel olarak sisteme girilir. Test adımlarını parametre girişinden oluştur butonuna basıldığında, okunmak istenen analiz dokümanı seçilmektedir. Ekrana girilen parametre, analiz dokümanında aranır. Dokümanda bulunan parametre, analiz dokümanının içeriği alanında kırmızı ile işaretlenir. Parametrenin içinde geçtiği cümleler, test adımı olarak listede gösterilmektedir. Elde edilen test adımları listesi Excel dokümanı olarak alınabilir.

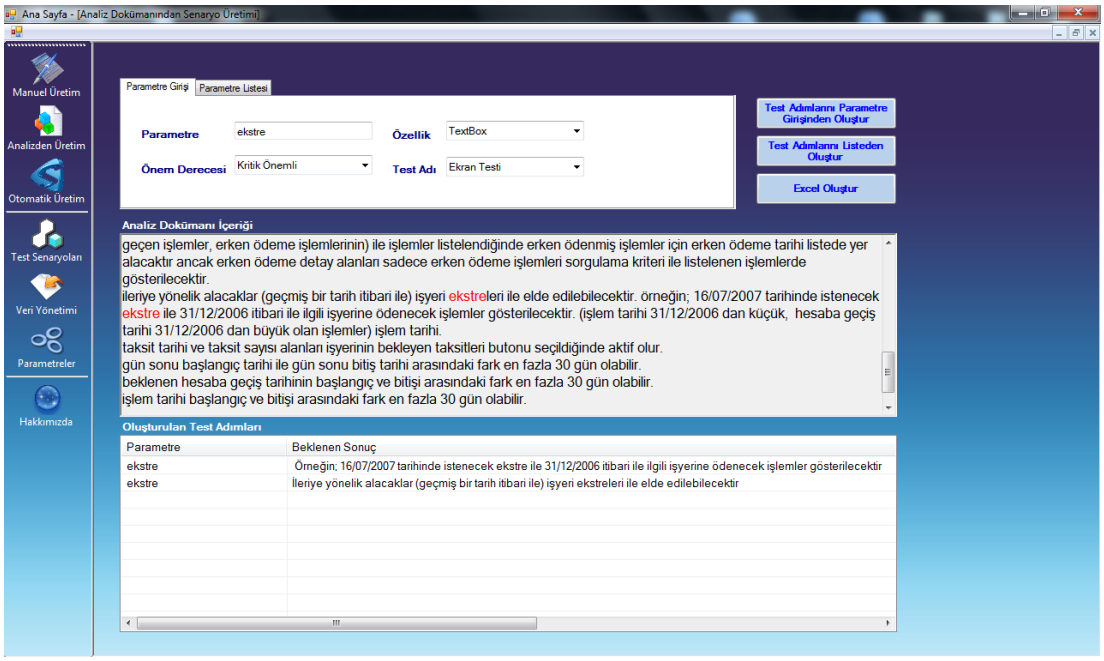
İkinci yöntemde ise, daha önce diğer ekranlardan girişi yapılan parametreler, liste şeklinde gösterilmektedir. Test adımlarını listeden oluştur butonuna basıldığında okunmak istenen analiz dokümanı seçilmektedir. Listede olan herbir parametre, okunan analiz dokümanında aranır. Dokümanda bulunan parametreler analiz dokümanının içeriği alanında kırmızı ile işaretlenir. Herbir parametrenin içinde geçtiği cümleler, test adımı olarak listede gösterilmektedir. Elde edilen test adımları listesi Excel dokümanı olarak alınabilir.



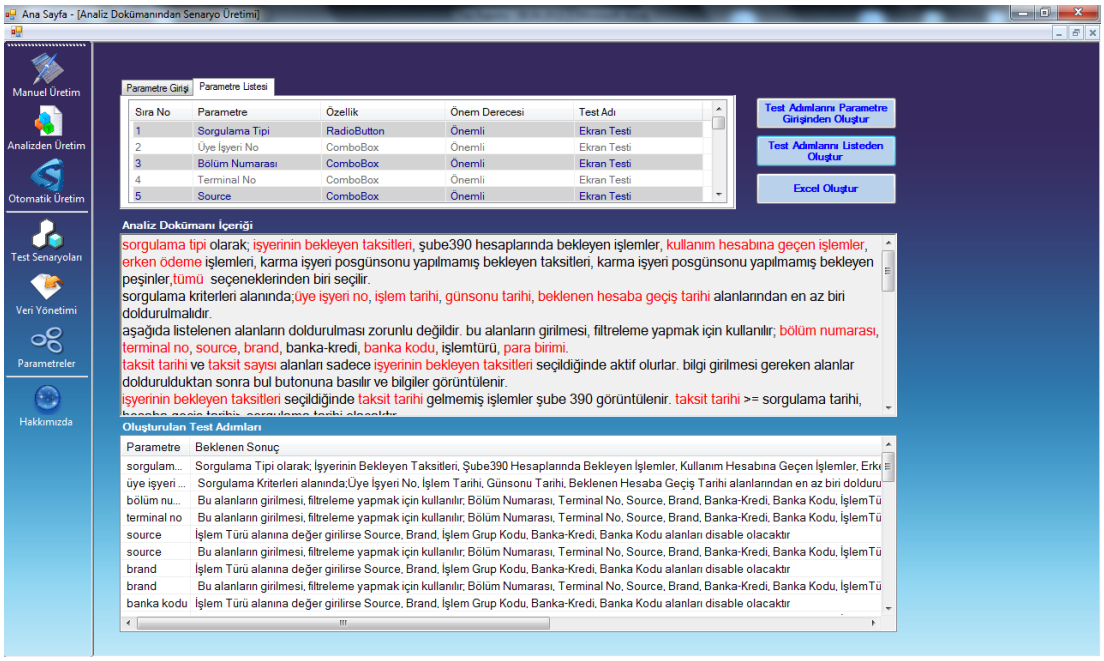
Şekil 4.9 Analizden üretim ekranı

Tablo 4.5 Analizden üretim ekranı görsel öğeleri

Öğe Adı	**Öğe Tipi**	**Durum**	**Zorunlu Alan**
Test Adımlarını Parametre Girişinden Oluştur	Buton	Aktif	Hayır
Test Adımlarını Listeden Oluştur	Buton	Aktif	Hayır
Excel Oluştur	Buton	Aktif	Hayır
Parametre Giriş Sekmesi	Sekme	Aktif	Hayır
Parametre Listesi Sekmesi	Sekme	Aktif	Hayır
Parametre	Metin Alanı	Aktif	Evet
Özellik	Seçim Listesi	Aktif	Evet
Önem Derecesi	Seçim Listesi	Aktif	Evet
Test Adı	Seçim Listesi	Aktif	Evet
Parametre Listesi	Liste	Aktif	Evet
Analiz Dokümanı İçeriği	Metin Alanı	İnaktif	Hayır
Oluşturulan Test Adımları	Liste	Aktif	Hayır



Şekil 4.10 Analizden üretim ekranı – Parametre girişinden üretim

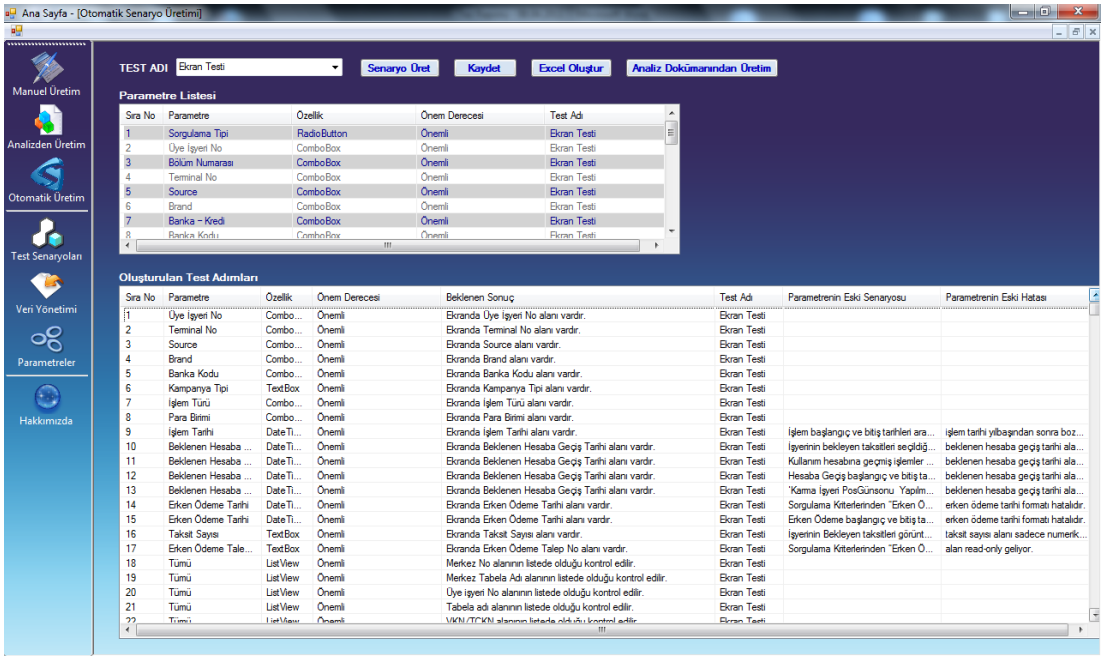


Şekil 4.11 Analizden üretim ekranı – Parametre listesinden üretim

4.4.4 Otomatik Senaryo Üretim Ekranı

Test edilmek istenen parametrelerin girişi, "Veri Yönetimi" ya da "Parametreler" ekranlarından yapıldıktan sonra "Otomatik Senaryo Üretim"

ekranında test adı kriterine göre getirilirler. Ekrandaki "senaryo üret" butonuna basıldığında, sistem, parametre listesindeki herbir kayıt için tek tek sorgulama yaparak, veri tabanında daha önce girişi yapılmış parametrelerle eşleşen bütün test adımlarını listede göstermektedir. Elde edilen test adımları listesi kaydedilebilmekte ya da Excel dokümanı olarak alınabilmektedir. Ayrıca bu ekrandan "Analiz Dokümanından Üretim" ekranına geçiş yapılabilmektedir.



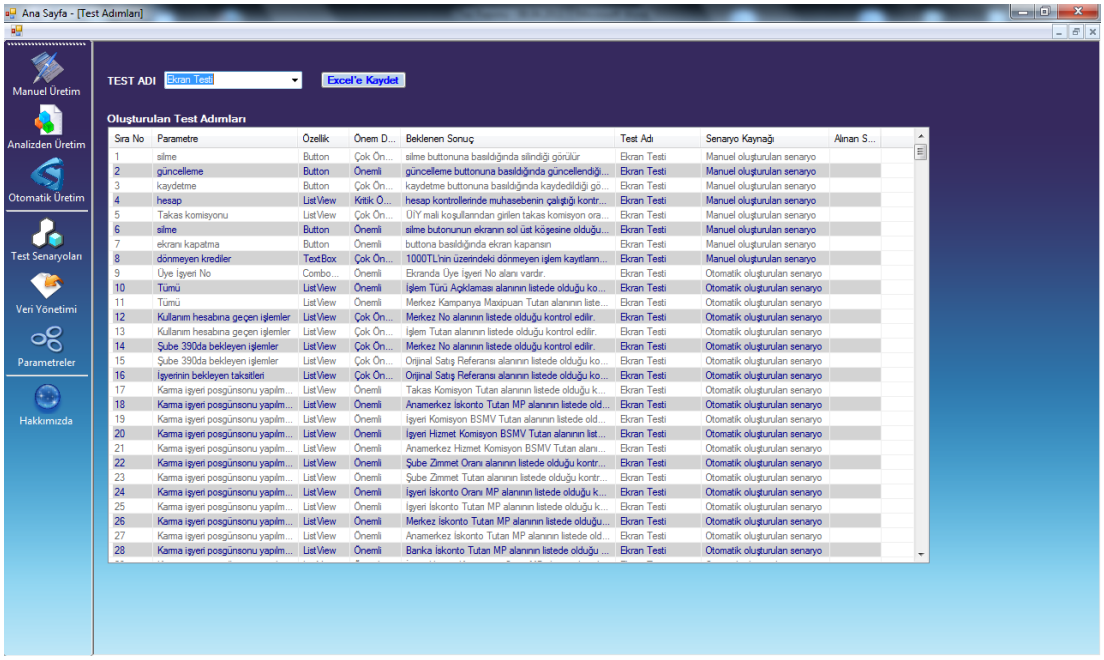
Şekil 4.12 Otomatik senaryo üretim ekranı

Tablo 4.6 Otomatik senaryo üretim ekranı görsel öğeleri

Öğe Adı	Öğe Tipi	Durum	Zorunlu Alan
Senaryo Üret	Buton	Aktif	Hayır
Kaydet	Buton	Aktif	Hayır
Excel Oluştur	Buton	Aktif	Hayır
Analiz Dokümanından Üretim	Buton	Aktif	Hayır
Test Adı	Seçim Listesi	Aktif	Evet
Parametre Listesi	Liste	Aktif	Evet
Oluşturulan Test Adımları	Liste	Aktif	Hayır

4.4.5 Test Senaryoları Ekranı

Sistemde oluşturulan ve kaydı yapılmış test adımları, ekran adı kriterine göre sorgulanarak listede gösterilmektedir. “Excel’e kaydet” butonuna basıldığında liste, Excel dokümanı olarak alınabilmektedir.



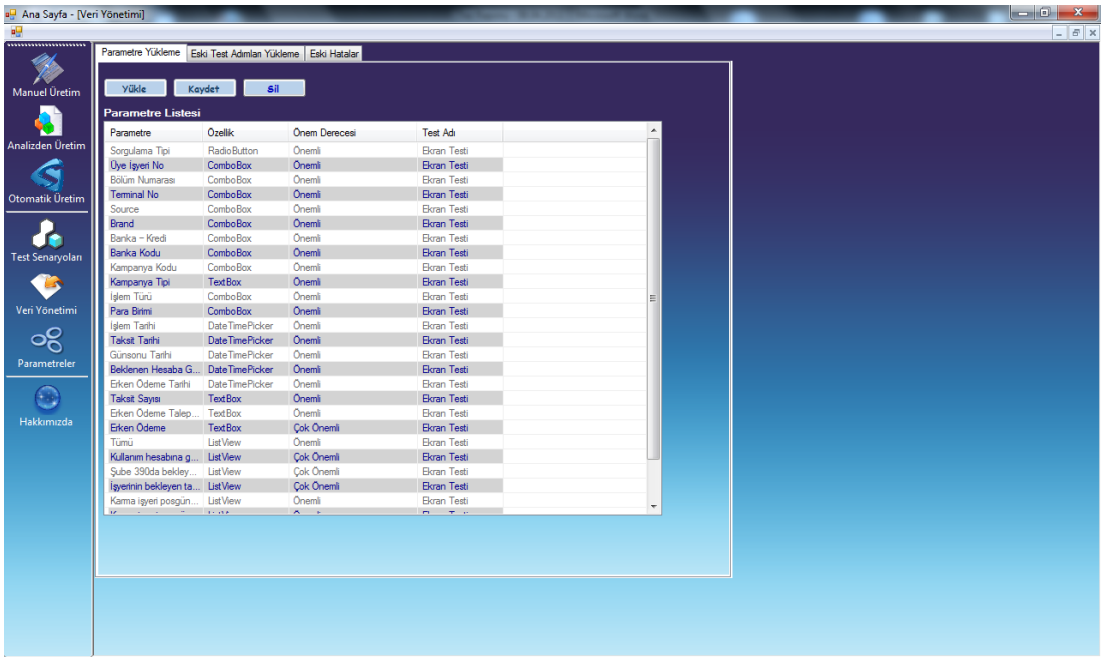
Şekil 4.13 Test senaryoları ekranı

Tablo 4.7 Test senaryoları ekranı görsel öğeleri

Öğe Adı	Öğe Tipi	Durum	Zorunlu Alan
Excel’e Kaydet	Buton	Aktif	Hayır
Test Adı	Seçim Listesi	Aktif	Evet
Oluşturulan Test Adımları	Liste	Aktif	Hayır

4.4.6 Veri Yönetimi Ekranı

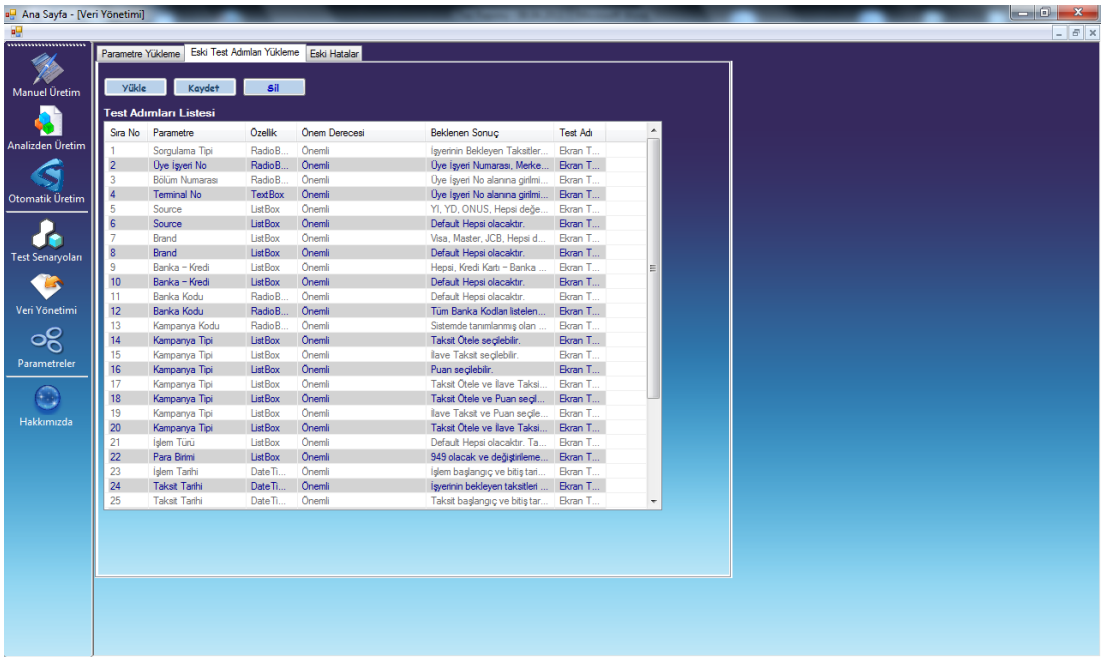
Bu ekran ile sisteme, Excel dokümanlarından toplu olarak veri girişi yapılmaktadır. Parametre, eski test adımları, eski hatalar sekmelerinde veri yükleme, kaydetme ve silme işlemleri yapılabilmektedir.



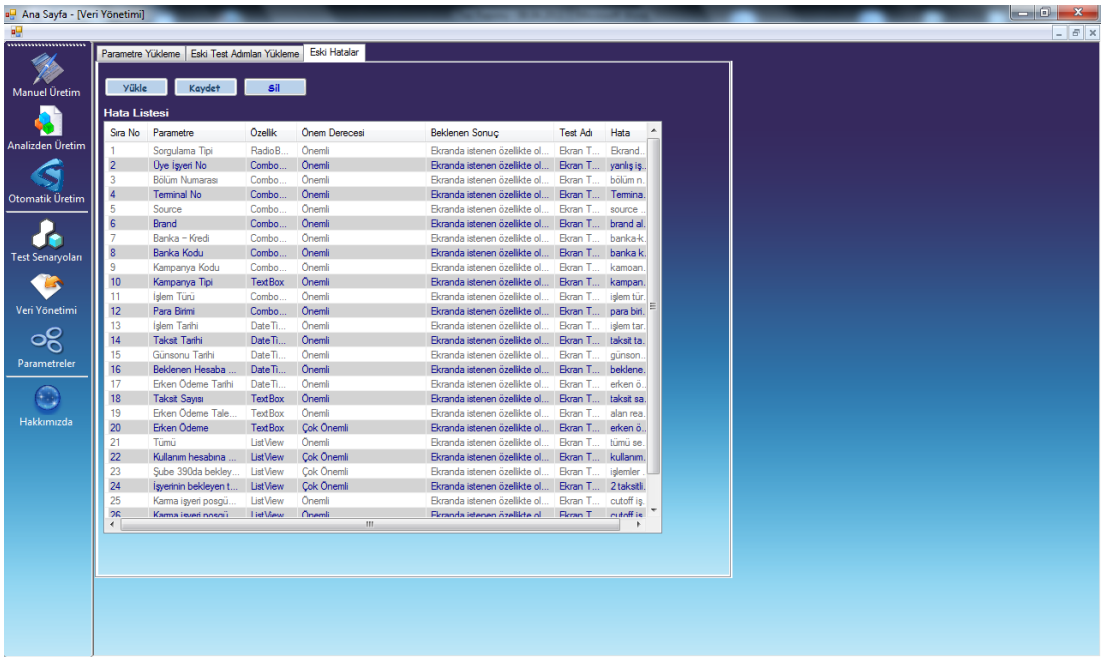
Şekil 4.14 Veri yönetimi ekranı – Parametre yükleme sekmesi

Tablo 4.8 Veri yönetimi ekranı görsel öğeleri

Öğe Adı	Öğe Tipi	Durum	Zorunlu Alan
Parametre Yükleme Sekmesi	Sekme	Aktif	Hayır
Eski Test Adımlarını Yükleme Sekmesi	Sekme	Aktif	Hayır
Eski Hatalar Sekmesi	Sekme	Aktif	Hayır
Yükle	Buton	Aktif/İnaktif	Hayır
Kaydet	Buton	Aktif/İnaktif	Hayır
Sil	Buton	Aktif/İnaktif	Hayır
Parametre Listesi	Liste	Aktif	Hayır
Test Adımları Listesi	Liste	Aktif	Hayır
Hata Listesi	Liste	Aktif	Hayır



Şekil 4.15 Veri yönetimi ekranı – Eski test adımları yükleme sekmesi



Şekil 4.16 Veri yönetimi ekranı – Eski hatalar sekmesi

4.4.7 Parametreler Ekranı

Bu ekranda, sistemde test edilmek istenen parametrelerin girişi, manuel olarak yapılmaktadır. Ayrıca ekranda, "Veri Yönetimi" ekranına erişim sağlanabilir.

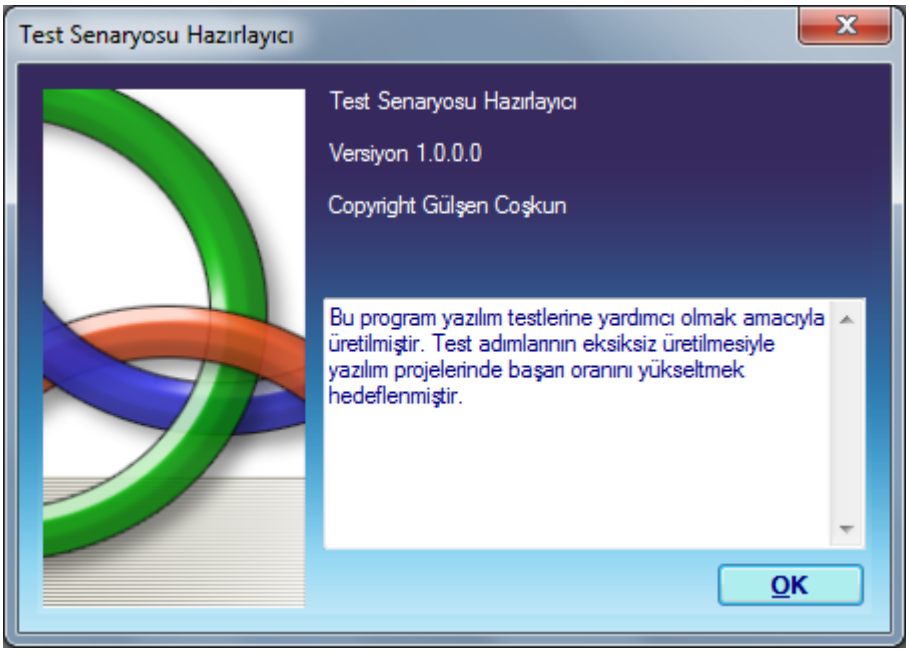
Şekil 4.17 Parametreler ekranı

Tablo 4.9 Parametreler ekranı görsel öğeleri

Öğe Adı	Öğe Tipi	Durum	Zorunlu Alan
Test Adı	Seçim Listesi	Aktif	Evet
Parametre	Metin Alanı	Aktif	Evet
Önem Derecesi	Seçim Listesi	Aktif	Evet
Özellik	Seçim Listesi	Aktif	Evet
Kaydet	Buton	Aktif	Hayır
Güncelle	Buton	Aktif	Hayır
Sil	Buton	Aktif	Hayır
Veri Yönetimi	Buton	Aktif	Hayır
Parametre Listesi	Liste	Aktif	Hayır

4.4.8 Hakkımızda Ekranı

Sistem bilgilerinin gösterildiği ekrandır.



Şekil 4.18 Hakkımızda ekranı

Tablo 4.10 Hakkımızda ekranı görsel öğeleri

Öğe Adı	**Öğe Tipi**	**Durum**	**Zorunlu Alan**
Bilgi Alanı	Yazı Alanı	İnaktif	Hayır
Açıklama Alanı	Metin Alanı	İnaktif	Hayır
OK	Buton	Aktif	Hayır

5. SONUÇLAR

Bu çalışmada, varolan sistemlerin incelenmesi sonucunda yeni bir sistem önerilmiş ve geliştirme yapılmıştır. Geliştirilen sistemin çalışma yapısı; kullanılan metotlar, veri tabanı tabloları, algoritmaların akış şemaları çizilerek anlatılmaktadır. Sistemi oluşturan modüllerin genel özellikleri ve kullanımlarıyla ilgili bilgilere, ayrı bir başlık altında yer verilmiştir. Ayrıca ekler bölümündeki bilgilerle sistemin ihtiyacı olmayan, yazılım testi ve yazılım kalitesiyle ilgili bilgiler bulunmaktadır.

Geliştirilen uzman yönlendirme sistemi, iş analizi ve test ekiplerinin hazırladığı test senaryolarının eksiksiz olmasını sağlamaktadır. Böylece yazılımda çıkabilecek her hata kontrol edilmiş olup, müşteri kullanmaya başladığında hata almaması sağlanacaktır. Geliştirilen sistemin uygulaması için örnek olarak, kredi kartları projelerindeki, testleri yapılmış, sorunsuz çalışan bir modül ele alınmıştır. Modülün, analiz dokümanı kullanılarak, ekranda kontrol edilmesi istenen her parametre belirlenmiştir. Geliştirilen sisteme girişi yapılan parametreler ile manuel, otomatik ve analiz dokümanı kullanılarak test senaryoları oluşturulmaktadır. Elde edilen test senaryolarının, testleri yapılacak ekranı baştan sona kontrol edebilecek derinlikte olduğu görülmüştür. Geliştirilen sisteme yüklenen veriler değiştirilerek, diğer sektörlerde kullanılan yazılım paketleri için de test senaryosu oluşturulabilir.

Proje kapsamında geliştirilen sistem ile başlangıçta belirlenen tüm hedefler gerçekleştirilmiştir. Sistemin geliştirileceği dil olarak, hakim olunan C# ve ortam olarak Microsoft Visual Studio seçilmiştir. Geliştirilen uzman yönlendirme sistemi, yazılım testleri için “birim test” ve “sistem testi” seviyesinde, test senaryolarını manuel ve otomatik olarak oluşturarak, iş analistleri ve test mühendislerinin, test senaryolarını oluşturma aşamasında ihtiyaçlarını karşılamaktadır. Modüler yapıda olduğu için sisteme yeni özellikler ekleme olanağı bulunmaktadır. Geliştirilmesi istendiğinde, yazılım sektöründe kullanılan test araçları seviyesine getirilebilir.

Bu tez çalışmasında, çalışma yaşamımda edindiğim bilgileri ve bugüne kadar almış olduğum dersleri etkin bir şekilde uygulayarak, iş yaşamında ihtiyaç duyulan projeyi geliştirme imkânına kavuştum.

KAYNAKLAR

[1] Kocatürk, M., Yazılım projelerinde başarı, 23.08.2010, http://www.kaynakkodlari.com/blog/yazilim-projelerinde-basari-38/.

[2] Silahtaroğlu, G. (2010), Sistem Analizi ve Tasarımı (1.Baskı), İstanbul: Papatya Yayıncılık.

[3] Bilgisite, Yazılım Kalitesi ve Test Metotları, 13.09.2010, http://www.bilgisite.com/yonetim/yonetim_05.htm.

[4] Rhxo, İş Analizi, 08.11.2010, http://www.rhxo.com.tr/surecler/is-analizi.html.

[5] İTÜ/BİDB, Yazılım Testi ve Test Süreçleri, 02.07.2010, http://www.bidb.itu.edu.tr/?d=1064.

[6] KEYTORC Yazılım Test Hizmetleri, ISQTB test eğitimi dokümanları, 06.06.2010.

[7] Ayrotek, Kalite ve Test Yönetimi, 2010, http://www.ayrotek.com.tr/urunler.html.

[8] Wikipedia, Boyer–Moore string search algorithm, 02.07.2010, http://en.wikipedia.org/wiki/Boyer-Moore.

[9] Boyer-Moore pattern matching, 11.08.2010, http://yasser-zamani.spaces.live.com/blog/cns!5AAB8D00414B403D!376.entry.

[10] Boyer, R. S., Moore, J. S., "A fast string searching algorithm". Communications of the ACM. 20:762-772, 1977.

[11] ŞEKER, Ş., Boyer Moore Dizgi Arama Algoritması, 2009, http://www.bilgisayarkavramlari.com/2009/05/19/boyer-moore-dizgi-arama-algoritmasi-boyer-moore-string-search/, 12.07.2010.

[12] Eralp, Ö., Case Araçları, 05.09.2010, http://www.yaztasarla.com/yazilim-muhendisligi/case-araclari.html.

[13] Microsoft, 06.12.2010, Microsoft Visual Studio 2010, http://www.microsoft.com/applicationplatform/tr/tr/Key-Technologies/Visual-Studio.aspx.

[14] Tayar, E., 08.10.2010, Kalite Risk Analizi, http://www.testgaraji.com/2010_03_01_archive.html.

[15] Carus A., Ersin A., Mesut A., 09.09.2010, "Dizgi Eşleme Algoritmalarının Alfabeye Bağlı Etkinliklerinin Araştırılması".

[16] Kilpel P., “Biosequence Algorithms Lecture 3: Boyer-Moore Matching”, 2005, http://www.cs.uku.fi/~kilpelai/BSA05/lectures/slides03.pdf, 11.12.2010, University of Kuopio Department of Computer Science.

[17] Kurbanoğlu, S., “Uzman Sistemler”, Türk Kütüphaneciliği 6, 4 (1992), http://tk.kutuphaneci.org.tr/index.php/tk/article/viewFile/1197/2393, 15.08.2010.

[18] “Yapay Zeka Bilgi İşlem Teknolojisi ve Bileşenleri” İktisadi ve İdari Bilimler Dergisi, Cilt: 14, Haziran 2000 Sayı: 1

[19] Bozdemir M., “Tasarımda Uzman Sistem Destekli Otomasyon Tekniklerinin Kullanımı”, Pamukkale Üniversitesi, Denizli 2004.
http://www.emo.org.tr/ekler/1e9abe7712be5fd_ek.pdf, 03.07.2010.

[20] Yıldız Teknik Üniversitesi, Lisans Dersleri, “Yapay Zeka Problem Çözme”, 10.09.2010, http://web.itu.edu.tr/~sonmez/lisans/ai/yapay_zeka_problem_cozme.pd

[21] Ersin K., “Dizgi eşleme algoritmalarinin incelenmesi ve yeni bir dizgi eşleme algoritması”, Trakya Üniversitesi, Edirne 2008.

ÖZGEÇMİŞ

Ad Soyad	Gülşen COŞKUN
Doğum Tarihi	23.06.1986
Doğum Yeri	Tekirdağ
Lise	Bahçelievler Dede Korkut Anadolu Lisesi
Lisans	Beykent Üniversitesi Bilgisayar Mühendisliği
Kariyer	Özel bir şirkette İş Analisti

EKLER

Ek-1: İş Analizi

Analiz evresinin temel amacı yeni kurulacak sistemden neler beklendiğini, bu sistemin ne yapması gerektiğinin ve sistemi kullanacak olan tüm bireylerin ne istediğinin araştırılarak ortaya çıkartılmasıdır. Bu evrede tüm işletmenin fonksiyon ve faaliyetleri incelenmeli, sistemin nasıl çalıştığı en ince ayrıntısına kadar araştırılmalıdır [1].

Analiz evresinde iş analistleri devreye girerek, müşteriler ve çözüm üretmeyi bilen sistem uzmanları arasında köprü vazifesi görürler. "İş Analisti" rolünün temel amacı, proje bünyesinde talep edilen uygulama değişikliklerinin hazırlık, analiz, test ve takibinden sorumlu olmakla birlikte ilgili proje kapsamında müşteri ile ilişkileri yürütmektir. İş analistleri, proje geliştirme sürecinin başarıyla tamamlanması içim kullanıcılardan alınması gereken bilgiyi verimli bir şekilde, değişik soru ve tekniklerle sağlarlar.

İş analistlerinin ana sorumlulukları aşağıdaki gibidir;

- Projelendirilen istek ile ilgili iş gereksinimlerini belirlemek ve analizini yapmak
  - Gereksinimlerin ölçülenebilir veya açıklıkla tanımlanabilir olmasını sağlamak.
  - Kullanıcıların ihtiyaçları doğrultusunda, sistem tasarım ve geliştirmelerinin yapılması için gerekli analizleri yapmak.
  - Kurumsal, sektörel ve teknolojik zorlukları/kısıtları kavrayarak, projeye konu olan işin güçlü ve zayıf yanlarını tespit etmek ve bunlara uygun çözüm önerileri getirmek.
  - İş gereksinimlerinin belirlenmesi ve gereksinimler üzerinde tüm tarafların mutabık kalması için gerekli toplantıları düzenlemek veya bu amaçla yapılan toplantılara katılmak, bilgi verici ve iyi organize edilmiş sunumlar hazırlamak.
  - Analiz dokümanlarını kalite standartlarına uygun olarak hazırlamak.
  - İş gereksinimi değişikliklerinde, önerilen çözüme uygun olarak, gereksinimleri, özellikleri ve iş süreçlerini gözden geçirerek düzeltmek ve yazılım geliştirme ekibine iletmek.
  - Teknik analiz ve geliştirme aşamasında ilgili teknik personele fonksiyonel tasarımın doğru anlaşılması için gerekli desteği vermek.
  - Proje yönetim ofisi ile koordineli bir şekilde çalışmak ve proje standartlarını uygulamak.
  - Proje maliyetinin belirlenmesi için ilgili gruplarla çalışma yapmak
  - Projenin, proje planına uygun olarak, zamanında ve planlandığı gibi yürütülmesine yardımcı olmak.

    - Projede üzerine atanan işleri, zamanında gerçekleştirmek, takip etmek ve raporlamak.
- Sistem ve kullanıcı kabul testlerinde test desteği vermek,
    - Sistem testlerinin planlama çalışmalarına katılmak, test senaryolarını hazırlamak ve test desteği vermek.
    - Kullanıcı kabul testlerinde test koordinatörlüğü görevini üstlenmek.
- Eğitim
    - Geliştirilen ürünlerle ilgili kullanım kılavuzlarını hazırlamak.
    - Proje paydaşlarının eğitim ihtiyaçlarını karşılamak.

Analistin, uygun bir çözüm önerebilmesi için iş uzmanları ile çalışması gerekmektedir. Daha sonra çözümü tasarlamak için teknik takım ile beraber çalışması gerekir. Bir çözüm önerisi; mevcut bir yazılımı, yeni yazılımı, süreç ya da iş akışı değişiklikleri ya da bunların hepsinin birleşimini içerebilir. Eğer çözüm dışardan yazılım satın almayı içeriyorsa, iş analistinin seçilen yazılımın iş ihtiyaçlarını karşıladığından emin olmak için iş uzmanları, IT kadrosu ve sağlayıcıları ile çalışması gerekmektedir. Bu süreçte, analistin detaylı iş ve işlevsel ihtiyaçları içeren teklif çağrısı hazırlaması gerekebilir. Eğer yeni bir yazılım kurmak ya da mevcut bir yazılımı yükseltmek gerekirse, analistin kullanıcı arayüzü, iş akışı tasarımı ve becerileri bildirmede yardım etmesi gerekir.

Yukarıda yazılanlarda belirtildiği gibi analist, yazılım projelerinin başarısında baştan sona, müşterinin isteklerini karşılayan çözümün çıkmasında, önemli bir rol oynamaktadır.

Ek-2: Kalite Risk Analizi

Yazılım söz konusu olduğunda akla gelen ilk olgu risklerdir. Yazılım testi, risklerin ortaya dökülmesi ve ölçeklendirilmesi ile başlar. Bu alandaki en önemli yöntem FMEA (Failure Mode Effect Analysis) dir [4].

Kalite Risk Analizi projede bulunan risklerin sınıflandırılarak öncelik kazandırılması işidir. Bunu yaparken esas amaç risk haritasının çıkarılmasıdır. Çıkarılan risk haritasının analiz edilerek proje süresince uygulanması, gereken test tipleri ile test çeşitlerinin ve yoğunluklarının belirlenmesi diğer amaçtır [4].

Kalite Risk Analizi hazırlanırken sınıflandırma önemli bir aşamadır. Kalite Risk sınıfları aşağıdaki gibidir.

Fonksiyonalite: Sistemin fonksiyonlarını tam olarak yerine getirememesine neden olabilecek risklerdir.

Yük & Kapasite: Maksimum sayıda kullanıcıyı, sistemin destekleyebilmesine izin vermeyecek risklerdir.

Güvenirlilik: Sistemin çalışamaz hale gelmesine sebep olabilecek risklerdir. Sistem kullanıcı açısında güvenli olmalıdır.

Sürdürülebilirlik: İşlemlerin devamlılığını kesintiye uğratabilecek risklerdir. Örneğin; yedekleme ve yeni sürümlerin yüklenmesi sırasında oluşabileceği düşünülen risklerdir.

Veri Kalitesi: Verilerin düzgün bir şekilde işlenmesi, saklanması ve çekilmesini engelleyebilecek risklerdir.

Performans: Normal şartlar altında işlemlerin belirlenen sürede yapılmasını engelleyebilecek risklerdir.

Dokümantasyon: Şirket personeli için hazırlanan kullanım klavuzu riskleridir. Klavuzun anlaşılır şekilde hazırlanması gerekmektedir.

Arayüz: Sistemin arayüzlerinde ortaya çıkabilecek risklerdir.

Entegrasyon: Sistemin diğer sistemlerle olan etkileşimi sırasında ortaya çıkabilecek risklerdir. Kalite risklerinin sınıflandırılmış durumu yukarıda anlatılmıştır. Bu risklerin ölçeklendirilmesi bir sonraki aşama olarak görülebilir. Her risk maddesinin önemi ya da olasılığı aynı değildir. Genelde firmalarda kullanılan şekli ile riske üç açıdan yaklaşılır [4].

Sistem açısından önemine göre, müşteri açısından önemine göre ve gerçekleşme olasılığına göre:

Sistem açısından önemine göre (severity);

Sistem açısından en önemliden az önemliye doğru sıralama yapılmıştır.

1. Veri Kaybı
2. İşlevsellik Kaybı
3. Giderilebilir İşlevsellik Kaybı
4. Kısmi İşlevsellik Kaybı
5. Kozmetik Riskler

Müşteri açısından önemine göre (priority);

Aşağıda müşteri açısından önem seviyelerine yer verilmiştir.

1. Acil
2. Zorunlu
3. Önemli
4. Düzeltilmesi İyi Olacak
5. İsteğe Bağlı

Gerçekleşme Olasılığına Göre (Likelihood);

1. Muhtemel
2. Mümkün
3. İhtimal Dahilinde Olmayan

Tüm risklerin derecelendirilmesiyle genel risk seviyesi bulunabilmektedir. Seviyenin sayısal gösterimi RPN (Risk Priority Number) ile sağlanır.

RPN = Severity x Priority x Likehood

(Örn1) Severitysi 2, Prioritysi 3 ve Likehood'u 2 olan bir risk için RPN Değeri; 2*3*2 = 12'dir.

(Örn2) Severitysi 1, Prioritysi 2 ve Likehood'u 1 olan bir risk için RPN Değeri; 1*2*1= 2'dir.

RPN değeri 2 olan risk, çok kritik risk olurken, RPN değeri 12 olan riskler, az kritik risk olarak değerlendirilecektir. Bu şekilde her bir riskin RPN değeri bulunduğunda FMEA çalışması tamamlanmış olmaktadır. Bu çalışma ile riskin ölçülebilmesi sağlanmaktadır. Hangi riske ne kadar adam/saat harcanacağı ve test süreci yönetilirken nelere ne kadar ihtiyaç duyulacağı belirlenmiş olmaktadır [4].

Ek-3: Yazılım Testi ve Test Süreçleri

Kaliteli yazılımlar, kabul edilebilir düzeyde hatalı, planlanan bütçe ile zamanında bitirilip dağıtılabilen, gereksinimleri ve beklentileri karşılayabilen ve sürdürülebilir özelliklere sahip yazılımlardır. Ancak, kalite terimi bağıl olup müşterisinin kim olduğuna ve tasarımda hedeflenen unsurlara bağlı olarak farklılıklar gösterebilmektedir. Doğal olarak, her kişinin kalite hakkında bireysel eğilimleri veya tercihleri söz konusu olmasına karşın kaliteyi ortaya koyan nesnel yöntemler yansız değerlendirmeleri olanaklı kılmaktadır [2].

Test, bir sistemi elle veya otomatik yollarla deneyerek, değerlendirerek, belirlenmiş gereksinimleri karşıladığının doğrulanması veya beklenen ile gözlenen sonuçlar arasındaki farkların belirlenmesi sürecidir. Yazılım testi ise bir yazılımın sonsuz sayıdaki çalışma alanından, sınırlı sayıda ve uygun şekilde seçilmiş testler ile beklenen davranışlarını karşılamaya yönelik, dinamik olarak yapılan doğrulama faaliyetlerini kapsamaktadır [2].

Testle ilgili genel algı, sadece testlerin çalıştırılması şeklindedir. Bu testin bir parçasıdır, fakat test aktivitesinin tamamı değildir. Test aktiviteleri testin çalıştırılmasından önce ve sonra da vardır. Örneğin, planlama ve kontrol, test durumlarının seçilmesi, test durumlarının tasarımı, sonuçların kontrolü, başarı kriterlerinin değerlendirilmesi, test süreciyle ve test edilen sistemle ilgili raporlama, testlerin sonlandırılması ve kapatılması. Test ayrıca dokümanların gözden geçirilmesini (kaynak kod dahil) ve statik analizi de kapsar [2].

Hem dinamik hem de statik test, benzer hedeflere ulaşmak için bir araç olarak kullanılabilir. Ayrıca, hem test edilen sistemin hem de geliştirme ve test süreçlerinin değerini artırmak için bilgi sağlamaktadır [2]. Aşağıdaki gibi farklı test hedefleri olabilir.

- Hata bulmak.
- Bilgi edinmek ve kalite seviyesi hakkında güvence sağlamak.
- Hataları engellemek.

Yaşam döngüsünün ilk aşamalarında test tasarlamak, hataların koda girmeden engellenmesine yardımcı olacaktır. Dokümanların gözden geçirilmesi de hataların koda girmesini engelleyici bir çalışmadır [2].

Testteki farklı bakış açıları, farklı hedefleri dikkate alır. Örneğin geliştirme testinde ana hedef, olabildiğinde çok hata üretip hataları ortaya çıkarmak ve çözülmesini sağlamaktır. Kabul testinde ana hedef, sistemin tasarlandığı gibi çalıştığının ve gereksinimlerin karşılandığına yönelik güvenin teyididir. Bazı durumlarda ise testin ana hedefi, her hangi bir zamanda yürürlükte olan yazılım risklerinin paydaşlara bilgi olarak verilmesi suretiyle (hataları düzeltme niyeti olmadan), yazılımın kalitesinin değerlendirilmesidir. Bakım testinin amacı, değişiklikler sırasında mevcut yazılıma yeni hatalar eklenmediğinin tespitidir. Operasyonel testin ana hedefi ise, güvenilirlik ve ulaşılabilirlik gibi sistem özeliklerinin değerlendirilmesidir [2].

Tablo Ek-1 Test terminolojisi

Terminoloji	**Açıklama**
Test Durumu	Belirli bir fonksiyonun testine yönelik, test girdileri, çalıştırma koşulları ve beklenen sonuçların bütünüdür.
Test Adımı	Testler esnasında yapılacak işlemi ve beklenen sonuçları açıklayan en küçük test yapısıdır.
Test Skripti	Otomatik testler esnasında, test araçları tarafından kullanılmak üzere hazırlanan programlardır.
Test Prosedürü	Bir test durumu süresince yapılacak işlemleri adım adım açıklayan dokümandır.
Test Aracı	Testin planlanması, yönetilmesi, gerçekleştirilmesi ve izlenmesi amacıyla kullanılan yazılımlardır.
Hata	Geliştirilen ürüne ilişkin özelliklerin karşılanmaması durumudur.

Ek-4: Yazılım Testinin Yapılma Nedenleri

Geliştirilen yazılımlarda, kaynağı ve sebebi değişmekle birlikte çeşitli hataların olması kaçınılmazdır. Çalışılan uygulama alanının kritikliğine göre bu hataların doğuracağı sonuçların biçimi değişebilmektedir. Bir finans uygulamasında yapılan küçük bir hata çok büyük miktarlarda para kaybına yol açabilecekken, bir askeri uygulamada yapılması

muhtemel küçük bir hata mal kaybının yanı sıra can kaybına da neden olma riskini taşımaktadır [2]. Bununla birlikte uygulamanın türüne bağlı olarak yazılım testleri,

- Müşteriye sunulmadan önce ürün kalitesinden emin olmak,
- Yeniden çalışma (düzeltme) ve geliştirme masraflarını azaltmak,
- Geliştirme işleminin erken aşamalarında yanlışları saptayarak ileri aşamalara yayılmasını önlemek, böylece zaman ve maliyetten tasarruf sağlamak,
- Müşteri memnuniyetini arttırmak ve izleyen siparişler için zemin hazırlamak,

amaçları doğrultusunda da yapılmaktadır. Ayrıca ürettikleri yazılımları yeterince test etmeden müşterilerinin hizmetine sunan firmalar, olası yazılım hataları sonucunda itibar kaybedecekleri gibi müşterilerine tazminat ödemek durumunda da kalabilirler [2].

Testin Yazılım Geliştirme, Bakım ve Operasyonlardaki Rolü

Sistemler ve operasyonlar yoğun olarak test edildiğinde tespit edilen hatalar sürüm öncesi çözüldüğünde, operasyonlar sırasında oluşacak problem riskleri azalır ve yazılım sisteminin kalitesi artar. Ayrıca yazılım testi, sözleşmesel ve yasal gereksinimler ile, endüstriye özgü standartların da karşılanmasını sağlamaktadır [3].

Test ve Kalite

Test, fonksiyonel ve fonksiyonel olmayan gereksinimler (kullanışlılık, güvenilirlik, sürdürülebilirlik vb) ve özellikler için tespit edilen hatalar yoluyla, yazılım kalitesinin ölçülmesine yardımcı olur. Test sırasında az sayıda ya da hiç hata bulunmaması, teste konu yazılımın kalitesi konusunda güvence sağlar. Düzgün tasarlanmış ve tamamlanmış testler, sistemin genel risk seviyesini azaltır.

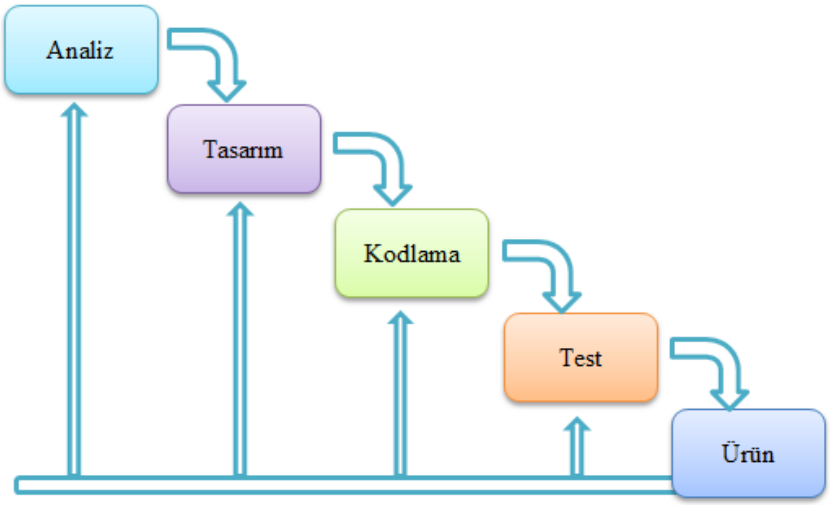
Test sırasında tespit edilen hataların çözümlenmesi, yazılım sisteminin kalitesini yükseltir. Ek olarak, önceki projelerden dersler çıkarılmalıdır. Diğer projelerdeki hataların nedenleri doğru olarak anlaşılırsa, süreçler geliştirilebilir ve aynı hataların tekrarı önlenebilir. Böylece gelecekteki sistemlerin kalitesi artırılabilir. Bu durum kalite güvencenin bir yüzüdür [3].

Yazılım Hatalarının Nedenleri

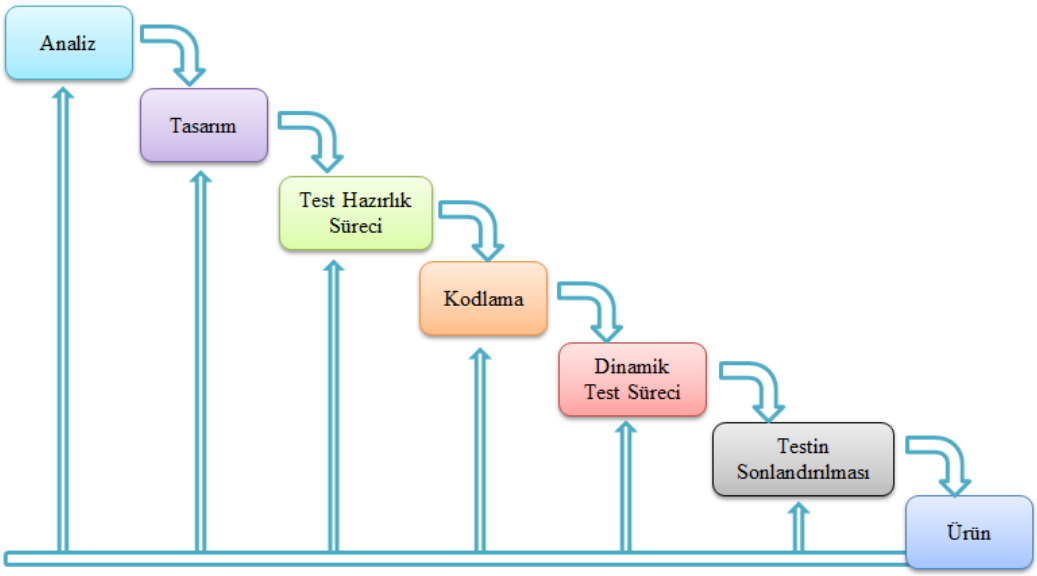
İnsanlar hata yaparlar, bu hatalar yazılımda, sistemde ya da dokümanda olabilmektedir. İnsanlar yanılabilir olduklarından, zaman baskısı altında çalıştıklarından, kodların ve alt yapının karmaşıklığından, değişen teknolojiler nedeniyle ya da bir den çok sistemin iç içe girmesi nedeniyle hatalar oluşur. ayrıca çevresel koşullar da donanımda arızalara neden olabilir [3].

Ek-5: Yazılım Test Süreci

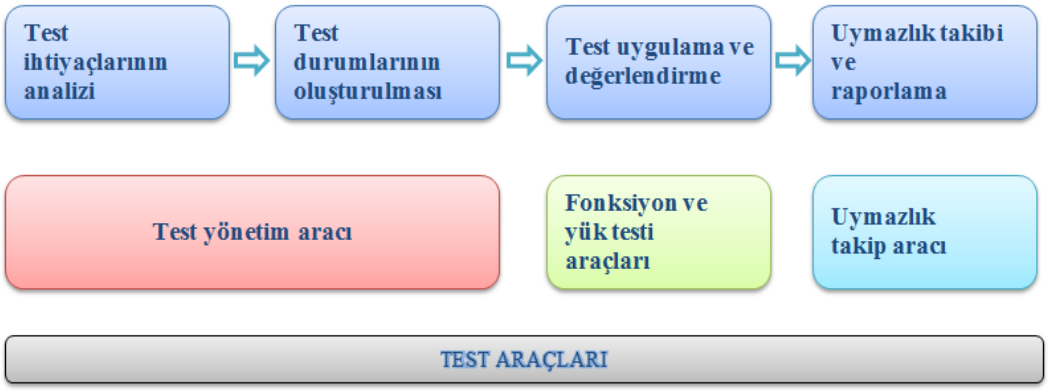
Genel olarak yazılım projeleri analiz, tasarım, kodlama, test, ürün süreçleri izlenerek geliştirilir. Bütün süreçler birbirini bu şekilde izlese de test süreci hiçbir zaman kodlama sürecinin bitmesini beklemez. İdeal bir yazılım test süreci, analiz, tasarım, test hazırlık süreci, kodlama, dinamik test süreci, testin sonlandırılması, ürün, şeklinde olmak durumundadır. Test süreçleri, aşağıdaki şekillerle ifade edilmiştir [2].



Şekil Ek-1 Genel süreç



Şekil Ek-2 İdeal süreç



Şekil Ek-3 Test süreci ve test araçları

Test İhtiyaçlarının Analizi

- Test Edilecek Sistem İhtiyaçlarının Analizi
- Geliştirim Metodolojisine Uygun Bir Test Yaklaşımının Oluşturulması
- Test Araçlarının Seçilmesi
- Test Ortamının Tasarlanması
- Test ve Değerlendirme Ana Planın Hazırlanması

Test Durumlarının Oluşturulması

- Test Ağacının Oluşturulması
- Test Durumları ile Sistem İhtiyaçlarının Eşleştirilmesi
- Test Planı Dokümanının Hazırlanması
- Test Senaryolarının Belirlenmesi
- Test Prosedürlerinin Yazılması
- Test Verisinin Belirlenmesi

Test Uygulama ve Değerlendirme

- Detaylı Test Uygulama Takviminin Hazırlanması
- Test Ortamının Kurulması
- Test Duyurusunun Yapılması
- Test Öncesi Brifinginin Verilmesi
- Testin Uygulanması
- Test Sonrası Brifinginin Yapılması
- Test Sonuçlarının Analiz ve Değerlendirmesi

Uymazlık (Hata) Takibi ve Raporlama

- Uymazlık Raporu (UR) Uygulama Esaslarının Belirlenmesi
- UR Durum İzleme Bilgilerinin Belirlenmesi
- Uymazlık Takibine Yönelik Sorumlulukların Belirlenmesi
- Uymazlık Takip Prosedürünün Hazırlanması
- Testler Esnasında Karşılaşılan Uymazlıkların Açılması
- Test Raporları Dokümanının Hazırlanması

Tablo Ek-2 Uymazlıkların (hataların) nitelikleri

Konu	**Bulunan Hataya İlişkin Konu Başlığı**
Tanım	Yakalanan hatanın detaylı olarak tanımlandığı bilgi alanıdır
Önem	Hatanın, sistem üzerindeki etkisini belirten, genellik ile 1-5 arasındaki sayılar
Öncelik	Hatanın ne zaman çözülmesi gerektiği bilgisini belirleyen, genellik ile 1-5 arasındaki sayılar
Statü	Hatanın hangi durumda olduğu bilgisi (statü bilgisi, uymazlıkların nasıl ele alınacağı ile ilişkilidir ve uymazlık takip prosedüründe tanımlanmalıdır)
Sürüm	Hatanın yakalandığı sürüm

Konfigürasyon	Hatanın yakalandığı, donanım ve yazılım konfigürasyonu
Çözüm	Çözüm için yapılan işlemlerin açıklandığı bölümdür. Çözümün yer aldığı arasürüm de belirtilmelidir.

Ek-6: Test Hazırlık Süreci

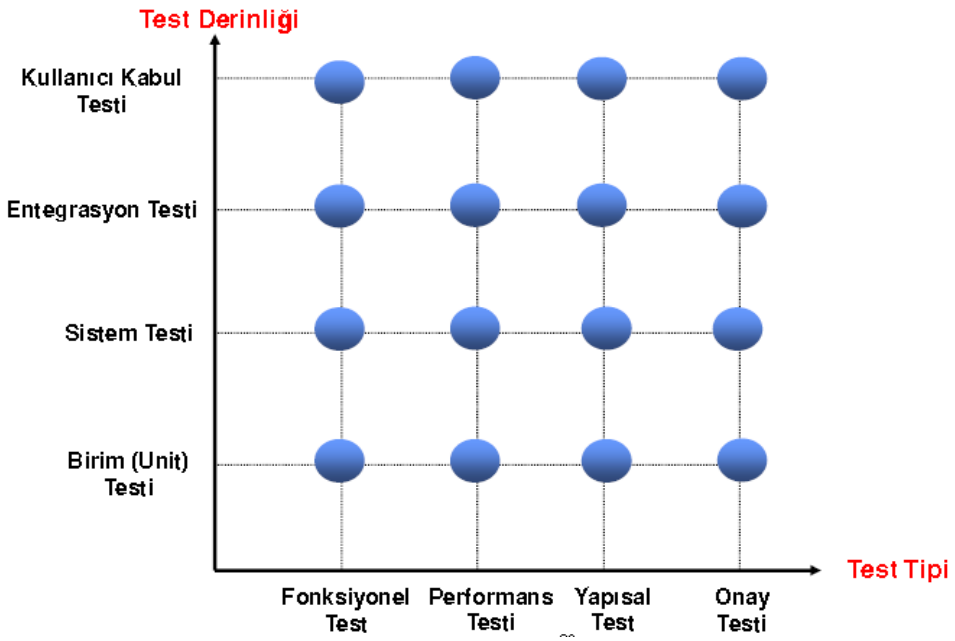
Bu süreç, yazılım test süreçleri içindeki ilk aşama olmakla beraber, testin efektif sonuçlar vermesi ve verimli olması açısından büyük öneme sahiptir. Dolayısıyla, bir yazılımı iyi test edebilmek için, test işlemlerinden önce, sağlıklı bir test hazırlık süreci kaçınılmazdır [2]. Test hazırlık sürecinde yapılması gereken birtakım standart işlemler şu şekilde sıralanır:

- Öncelikle test edilecek yazılıma ait analiz ve teknik tasarım aşamaları ile ilgili dokümanlar, test ekibi tarafından incelenir.
- Yazılım içinde test edilecek ve edilmeyecek modüller belirlenir.
- Risk analizi yapılır ve yapılan değerlendirmeye göre dinamik test aşamasında uygulanacak olan test teknikleri ve metodları belirlenir.
- Dinamik testin uygulanacağı ortamlar ve bu ortamların ihtiyaçları belirlenip, uygun şartlar sağlanır.
- Test ekibi içinde görev paylaşımı ve zaman planlaması yapılır.
- Testin sonlandırma kriterleri belirlenir.
- Bir programa belirli girdiler (input) verildiğinde hangi çıkışların (output) ne şekilde alınması gerektiğini bildiren test durum senoryaları belirlenir.
- Dinamik testin hangi adımlarla ve ne şekilde uygulanacağının belirtildiği test planı hazırlanır.

Yukarıda sıralanan test hazırlık sürecine ait aşamalar gerçeklendikten sonra dinamik yazılım testi aşamalarına geçilmektedir.

Ek-7: Dinamik Test Süreci

Dinamik test süreci, kodlama çalışmalarının bitmesine yakın bir dönemde başlar. Bulunan hatalar çözüldüğünde, testin sonlandırma kriterleri sağlanarak, test sona erer. Test edilecek yazılımın türüne göre, dinamik olarak uygulanacak test teknikleri ve bu tekniklerin uygulanma metotları farklılık gösterebilir [2].



Şekil Ek-4 Test derinliği ve tipleri

Genel olarak dinamik test süreci içinde ve sonrasında uygulanabilecek olan testler ve test teknikleri şu şekilde sıralanır:

- **Birim Testi (Unit testing) :** Dinamik test sürecinin ilk aşaması olmakla beraber, hataların erken bulunup düzeltilebilmesi açısından bu sürecin en önemli kısmını oluşturur. Mikro ölçekte yapılan bu testte, özel fonksiyonlar veya kod modülleri (fonksiyonlar, prosedürler, nesneler vb.) test edilir. Bu test, test uzmanlarınca değil, programcılar tarafından yapılır. Program kodunun ayrıntıları ile içsel tasarım biçiminin bilinmesi gerekir. Uygulama kodu çok iyi tasarlanmış bir mimaride değilse oldukça zor bir testtir [2].

- **Entegrasyon Testi (Integration testing) :** Bir uygulamanın farklı yazılım modüllerinin beraberce uyum içinde çalışmasını sınamak için yapılan bir testtir. Bileşenler, modüller, bağımsız uygulamalar, istemci/sunucu uygulamaları biçiminde olabilmektedir. Bu tür

testlere, özellikle istemci/sunucu uygulamaları ve dağıtık sistemlerin testinde başvurulmaktadır. Bunun yanısıra uygulamaya yeni işlevsel elemanlar ya da program modülleri eklendikçe sürekli test edilmesi işlemine de "Artımsal Tümleyim Testi" adı verilir. Test uzmanları veya programcılar tarafından gerçekleştirilen testlerdir [2].

- **Regresyon Testi (Regression testing) :** Uygulamada ve uygulama ortamlarında gerekli değişiklikler ve sabitlemeler yapıldıktan sonra yeniden yapılan testlere regresyon testi denilir. Böylece, önceki testlerde belirlenen sorunların giderildiğinden ve yeni hatalar oluşmadığından emin olunur. Uygulamanın kaç kez yeniden test edilmesi gerektiğini belirlemek güçtür. Bu nedenle, özellikle uygulama geliştirme döneminin sonlarına doğru yapılır [2].

- **Yük – Performans Testi (Performance testing) :** Beklenmedik (normal olmayan) ağır yükler, belirli eylemler ve taleplerin çok fazla artışı, çok yoğun sayısal işlemler, çok karmaşık sorgulamalar vb. ağır koşullar altında olan bir sistemin fonksiyonel testi (iş yapabilme testi) olarak da tanımlanabilmektedir. Bir web sitesi için sistem tepkisinin hangi noktada azaldığı veya yanıt veremez olduğunu belirlemek için yapılan testler, performans testine örnek teşkil edebilir [2].

- **Kullanıcı Kabul Testi (User acceptance testing) :** Son kullanıcı veya müşteri siparişine (veya isteklerine) dayanan son test işlemidir. Kullanıcıların, uygulamayı kabul etmeden önce, söz konusu uygulamanın gereksinimlerini ne ölçüde karşılayıp karşılamadığını belirleyip, geri dönüş yapabileceği testlerdir. Kabul testinin çeşitleri aşağıdaki gibidir [2].

  Kullanıcı kabul testi : İş kullanıcıları tarafından, sistemin kullanımının uygunluğunu doğrular [2].

  Operasyonel (kabul) testi: Sistem yöneticileri tarafından sistemin kabulü. Bu tarz bir test aşağıdakileri içerir:

  - Yedekleme/geri yükleme testi (testing of backup/restore);
  - Yıkım onarımı (disaster recovery);
  - Kullanıcı yönetimi (user management);
  - Bakım görevleri (maintenance tasks);
  - Güvenlik duyarlılıkları ile ilgili periyodik kontroller (periodic checks of security vulnerabilities).

  Sözleşme ve düzenleme kabul testi (Contract and regulation acceptance testing): Sözleşmeye dayanarak geliştirilen yazılımın, sözleşmede yer alan kabul kriterine göre kabul testinin yapılmasıdır [2].

Alpha ve beta testi (Alpha and beta testing): Pazar geliştiricileri, yazılım ürünlerini ticari olarak satışa sunmadan önce var olan ya da potansiyel müşterilerinden geri bildirim almak isterler [2].

- Alpha testi, geliştirici örgütün sitesinde gerçekleştirilir.
- Beta ya da alan testi, müşterilerin kendi mekanlarında gerçekleştirilir.

Her iki test de, ürünün geliştiricileri tarafından değil, potansiyel müşterilerce gerçekleştirilir [2].

- **Beyaz Kutu Test Tekniği (White box testing technic) :** Beyaz kutu test tekniğinin en genel tabiri kod testidir. Projenin hem kaynak kodu, hem de derlenmiş kodu test edilir. Bu tür testler, uygulama kodunun iç mantığı üzerindeki bilgiye bağlıdır. Yazılım kodundaki deyimler, akış denetimleri, koşullar vb. elemanlar sınanır [2].
- **Kara Kutu Test Tekniği (Black box testing technic) :** Test ekipleri tarafından en çok kullanılan teknik olan kara kutu test tekniği, adından da anlaşılacağı gibi uygulamanın sadece derlenmiş kodu üzerinden test edilmesi olarak bilinmektedir. Bu test tekniğinde, yazılımın yapısı, tasarımı veya kodlama tekniği hakkında herhangi bir bilgi olması gerekli değildir. Yazılımın, müşterinin isteğine yanıt verip veremediği ve işlevselliği sınanmaktadır [2].

Ek-8: Testin Sonlandırılması

Yapılan testler sonucunda bulunan hatalar düzeltildikten sonra test sonlandırma kriterleri (test hazırlık süreci) kontrol edilir. Eğer tüm kriterlerin kabul edilebilir düzeyde sağlandığı tespit edilirse test sonlandırılır. Testin sonlandırılmasının ardından uygulama, müşteri testine verilir (Kullanıcı kabul testi). Müşterilerin bulduğu hatalar veya değiştirilmesi istenilen noktalar gözden geçirilerek tekrar test ekibinin kontrolüne sunulur. Bu kontrolden çıkan uygulama, ürün aşamasına geçer. Böylece yazılım test süreci sona erdirilerek, yazılım geliştirme sürecinin son basamağına geçilmiş olunur [2].

EK KAYNAKLARI

[1] Rhxo, İş Analizi, 2010, http://www.rhxo.com.tr/surecler/is-analizi.html.

[2] İTÜ/BİDB, Yazılım Testi ve Test Süreçleri, 2010, http://www.bidb.itu.edu.tr/?d=1034.

[3] KEYTORC Yazılım Test Hizmetleri, ISQTB test eğitimi dokümanları, 2010.

[4] Tayar, E., 2010, Kalite Risk Analizi,

http://www.testgaraji.com/2010_03_01_archive.html.