TEST OLGUNLUK MODELİ ENTEGRASYONU (TMMi) İLE YAZILIM TEST SÜREÇLERİNİN İYİLEŞTİRİLMESİ

Gökhan ŞİT

YÜKSEK LİSANS TEZİ
TEKNOLOJİ VE İNOVASYON ANABİLİM DALI

Danışman
Doç. Dr. M. Burak BİLGİN

AMASYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HAZİRAN 2019

ETİK BEYAN

Amasya Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

İmza:
Adı Soyadı: Gökhan ŞİT
Tarih:

TEST OLGUNLUK MODELİ ENTEGRASYONU (TMMi) İLE YAZILIM TEST SÜREÇLERİNİN İYİLEŞTİRİLMESİ

(Yüksek Lisans Tezi)

Gökhan Şit

AMASYA ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Haziran 2019

ÖZET

Bu çalışmada test süreçlerinde iyileştirme anlayışına sahip yazılım sektöründe faaliyet gösteren firmalardan Test Olgunluk Modeli Entegrasyonu (TMMi) 2. ve 3. seviye değerlendirmelerini, denetlemeden önce kendi içlerinde yapabilmelerine yönelik geliştirilen olgunluk seviye belirleme ve değerlendirme yöntemi verilmiştir. Bu yöntem ile şirketlerin yüksek bütçe ayırarak yaptıracakları değerlendirmelere katılmadan önce kendi öz değerlendirmelerini yapabilmeleri ve olgunluk seviyesini artırarak test süreçlerinde iyileştirme yapabilmelerine yardımcı olmak amaçlanmıştır. Bu yöntemin geçerliliği uygulamalı olarak TMMi modelinin 2nci ve 3üncü seviyeleri değerlendirmelerinde sınanmıştır. Yazılım test alanında bulunan şirketler, yazılım test süreçlerini daha etkili şekilde anlayıp geliştirmelerine yardımcı olmak için bu tez kapsamında hazırlanan test olgunluk seviyesi belirleme yöntemini kullanarak isterlerse resmi bir TMMi denetimine hazırlık yapabilirler. Bu şirketler isterlerse süreçlerini iyileştirerek daha kaliteli ve hatasıza yakın ürünler oluşturabilir ya da TMMi sertifikasına sahiplerse bunun devamlılığını sağlamak için istedikleri zaman kendilerini denetleyebilirler. Bu çalışmayla birlikte, yazılım test süreci ile ilgili TMMi modeli kullanımının gösterilmesi, test süreç olgunluk seviyesinin belirlenmesinde rehber olabilecek öneride bulunulması, literatürde az sayıda mevcut kaynak bulunan TMMi konularında ayrıntılı bir çalışma elde edilerek literatüre katkı sağlamak ve pratik uygulama içermesi nedeniyle firmaları rahatlatacak bir çalışma olması önemli katkılar olarak görülmektedir.

SayfaAdedi :141
AnahtarKelimeler : Yazılım test süreçleri, test süreç olgunluk seviye belirleme, yazılım test süreçlerinin iyileştirilmesi, TMMi model yaklaşımı, yazılım geliştirme süreçlerinde test
Danışman: :Doç. Dr. M. Burak BİLGİN

IMPROVEMENT OF SOFTWARE TESTING PROCESSES WITH TEST MATURITY MODEL INTEGRATION (TMMi)

(M.Sc. Thesis)

Gökhan Şit

AMASYA UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

June 2019

ABSTRACT

In this study, maturity and level determination and evaluation methods which are developed in order to be able to perform TMMi 2nd and 3rd Level evaluations before the audit are given from the firms operating in the software insdustry with the understanding of improvement in test processes. With this method, it is aimed to help companies to make their own self-assessments before they participate in the evaluations that they will make by allocating high budget and to improve the test processes by increasing their maturity level. The validity of this method has been tested in the 2nd and 3rd level evaluations of TMMi. Companies operating in the software industry can prepare for an official TMMi audit if they wish, using the test maturity level determination method prepared within the scope of this thesis, to help them better understand and develop software testing processes. If they want to improve their processes, they can create better quality and valid products, or if they already have such a certificate, they can control themselves at any time to ensure its continuity.With this study, it will be a study that will be helpful to show the usage of TMMi model related to the software testing process, to make a suggestion that can be a guide in determining the test process maturity level, to make a contribution to the literature by taking a detailed study on TMMi subjects which have a small number of available resources in the literature and to include practical application.

PageNumber :141
KeyWords : Software test processes, test process maturity level, software test processes improvement, TMMi model approach, software development process testing
Supervisor :Doç. Dr. M. Burak BİLGİN

ÖN SÖZ ve TEŞEKKÜR

Bu çalışmada test süreçlerinde iyileştirme anlayışına sahip yazılım sektöründe faaliyet gösteren firmalardan TMMi 2nci ve 3üncü seviye değerlendirmelerini, denetlemeden önce kendi içlerinde yapabilmelerine yönelik geliştirilen olgunluk seviye belirleme ve değerlendirme yöntemi verilmiştir. Bu yöntem ile şirketlerin yüksek bütçe ayırarak yaptıracakları değerlendirmelere katılmadan önce kendi öz değerlendirmelerini yapabilmeleri ve olgunluk seviyesini artırarak test süreçlerinde iyileştirme yapabilmelerine yardımcı olmak amaçlanmıştır. Bu yöntemin geçerliliği uygulamalı olarak TMMi modelinin 2nci ve 3üncü seviyeleri değerlendirmelerinde sınanmıştır.

Bu tez çalışmasında desteklerini esirgemeyen tez danışmanım Doç. Dr. M. Burak BİLGİN' e, Amasya Üniversitesi UZEM müdür yardımcısı Öğ. Gör. Sabri Serkan TAN' a, tez çalışması boyunca beni yönlendiren Tahir EMİRHAN' a, Okan SARICA' ya ve İsmail Serkan BAKIRCI' ya teşekkürü borç bilirim. Ayrıca tez dönemi boyunca benden desteklerini hiçbir zaman esirgemeyen aileme ve çok sevdiğim nişanlıma teşekkür ederim.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

Sayfa

Sayfa

ÇİZELGELER DİZİNİ

Çizelge **Sayfa**

Sayfa

ŞEKİLLER DİZİNİ

Şekil **Sayfa**

Sayfa

SİMGELER ve KISALTMALAR DİZİNİ

Bu çalışmada kullanılmış bazı simgeler ve kısaltmalar açıklamaları ile birlikte sunulmuştur.

Kısaltmalar	Açıklama
TMMi	Test Olgunluk Model Entegrasyonu
CMMI	Yetenek Olgunluk Model Entegrasyonu
SP	Özel Uygulama
SG	Özel Hedef
TPCR	Test Süreç ve Yetenek Derecelendirmesi
ISTQB	Uluslararası Yazılım Test Kalite Alanı
CKSY	Cihaz Kiralama ve Sipariş Yönetimi
TD	Test Durumu
TMA	Test Olgunluk Değerlendirmesi
TPI	Test Süreci İyileştirme
CMMI-SVC	Servisler için Uyumluluk Olgunluk Modeli Entegrasyonu
PRİSMA	Pratik Risk Bazlı Test
TIM	Test Geliştirme Modeli
TOM	Test Organizasyonu Olgunluk Modeli
BDTM	İş Odaklı Test Yönetimi
SEI	Software Engineering Institude

1.GİRİŞ

Günümüzde birçok şirket, yazılımın ve genel iş süreçlerinin kalitesini sağlamak için test sürecini iyileştirmenin gerekli olduğunu fark eder. Ancak, uygulamada süreci iyileştirmek ve kontrol etmek için gereken adımları tanımlamak zor olabilmektedir. Yazılım sektöründe gelişmelerin ve değişikliklerin çok hızlı gerçekleşmesi ve verimliliğin sürekli artması ile müşteriler daha yüksek kalite seviyeleri talep etmektedirler. Mevcut test süreci tatmin edici olsa bile, bu durumun gelecekte iyileştirilmesi gerekecektir. Şirketler test süreçlerini gözden geçirmeli ve proaktif ve ileri görüşlü olmak durumunda kalmaktadırlar. Müşterilerin ihtiyaçlarını olduğu kadar kendi ihtiyaçlarını da karşılamak için sürekli gelişen test için tanımlanmış bir standarda veya sürekli iyileştirme programına sahip olmayı tercih ederler. Süreç iyileştirme programına sahip olmak isteyen ya da mevcut süreçlerini değerlendirmek isteyen şirketler danışmanlık maliyeti ödeyerek şirketlerindeki test süreç iyileştirme çalışmaları için ek maliyetlere katlanmak durumunda kalmaktadırlar. Ekstra maliyet olacak bu çalışmalara girmek istemeyen şirketler ise yazılım geliştirme süreçlerinde testin etkin olarak kullanılmasından çok çalışanlarının inisiyatiflerine bırakarak yazılım ürünlerindeki çıkacak hatalara ve bu hatalardan doğacak maliyetlere girmektedirler. Tez çalışmasında test olgunluk seviyesi belirleme ölçümü için öneride bulunulmuş ve seviye yükseltmek için literatürde kullanılan modeller verilmiş ve bu modellerden Test Olgunluk Model Entegrasyonu (TMMi) ayrıntılı olarak ele alınmıştır.

1.1. Literatür Taraması

2017 yılında Alparslan tarafından yapılan çalışmada yazılım sektöründe bulunan şirketlere kaliteli ürün oluşturmada süreç yönetiminin öneminden ve kaliteden bahsedilmiş ve bu konu ile ilgili uygulanan yöntemleri ve modelleri göstererek Yetenek Olgunluk Model Entegrasyonu (CMMI) detaylı olarak ele alınmıştır. Çalışmasında CMMI 3. olgunluk seviyesinde yetkinliği ölçecek bir envanter sunmuş ve sektörde faaliyet gösteren firmalara uygulanarak geçerliliğini sınamıştır. Bu envanterin kullanıldığı firmalardan biri ile mülakat tekniği ile görüşme yaparak sertifikasyon sürecindeki deneyimler ve sağladıkları faydalar yazılım sektöründeki firmalara örnek olması adına incelemiştir [1].

2006 yılında Gül tarafından yapılan çalışmada anket uygulaması yaparak yazılım geliştirme süreçlerinde iyileştirme yapan şirketlere ilişkin bulguları değerlendirmiş ve iyileştirilmesi gereken durumları saptamıştır. Türkiye'deki şirketlerin yazılım geliştirme sürecini, bu süreçte karşılaşılan sorunları, yapılan iyileştirme çalışmalarını, kullanılan teknik ve araçları ve yazılım projelerinin başarı kriterlerine göre durumlarını anket çalışması ile vermiştir [2].

Garousi, Felderer ve Hacaloğlu 2017 yılında yaptıkları çalışmalarında, test olgunluk değerlendirmesi (TMA) ve test süreci iyileştirme (TPI) yaklaşımlar ve çerçeveleri konusunda detaylı incelemeler yapmışlar ve önerilerde bulunmuşlardır. 58 farklı test olgunluk modeli ve bu konu hakkında çeşitli derecelerde deneye dayalı kanıtlara sahip çok sayıda kaynağı tespit etmişlerdir. Test süreçlerinin olgunluğunu değerlendirmek ve geliştirme konusunda alanın en kapsamlı araştırma yaparak bir derleme yapmışladır [3].

Suadarsanam 2013 yılındaki çalışmasında, her bir olgunluk seviyesi için TMMi modelinin süreç alanlarını derecelendirmek için bir metodoloji sunmuştur. Süreç özellik derecelendirme ölçeğinde bulunan dört ölçü ve TMMi düzeylerinin her biri için işlem uyumluluğunu ölçmek için niceliksel bir yöntem sunulmuştur [4].

Sarıalioğlu ve Dülger tarafından 2011 yılında yaptıkları çalışmalarında, test süreçlerinin olgunluk seviyesi ölçümü için kullanılan modelleri vermişlerdir. Bu modeller içerisinden Test Süreç ve Yetenek Değerlendirmesi (TPCR)'in yapısı ve özelliklerine yer vermişlerdir. Mevcutta bulunan modeller ile henüz çözülmemiş sorunların belirlenmesi ve test olgunluk seviyesinin doğru şekilde belirlenmesi bu model ile hedeflenmiştir [5].

Arkanın 2016 yılındaki çalışması, yazılım test hizmetleri sunan çeşitli endüstri ortaklarının önerileri ve Servisler için Uyumluluk Olgunluk Modeli Entegrasyonu (CMMI-SVC) modeli temel alınarak geliştirilen Yazılım Test Servisleri için Olgunluk Modelini içermektedir. Çalışmasında modelin uygulanabilirliğini ve yararlılığını değerlendirmek için Türkiye'de yazılım test hizmetleri sağlayan iki şirkette uygulayarak endüstriyel uygunluğu değerlendirmiştir. Bu vaka çalışmasının nitel ve nicel sonuçları, önerilen modelin test yöneticilerine ve test hizmeti sağlayan firmalara yardımcı olduğunu göstermiştir. Bu model ile yazılım test endüstrisinin ihtiyaçlarını karşılama amaçlanmış, tamlık ve uygunluk gibi iki kriter üzerinde durduğu görülmüştür [6].

Serdaroğlu tarafından 2015 yılında yapılan yazılım test süreci için kullanılan Test Olgunluk Model Entegrasyonu (TMMi) ve Pratik Risk Bazlı Test (PRISMA) tekniklerinin anlaşılmasını amaçladığı bu çalışmada ISTQB(International Software Testing Board) rehber dokümanını özetlemiştir. Bu çalışmada web tabanlı geliştirilen bir sistem üzerinde TMMi modeli 2. seviyede uygulanmıştır ve TMMi modelinin test planlama safhasında PRISMA yaklaşımı gösterilmiştir. Sonuç olarak risk odaklı test yaklaşımı ile test sürecinde riskli olan alanlarla ilgili hatalar daha önce tespit edilmiştir. Bu durum yazılım geliştirme projesinde fazla gecikmeler olmasının önüne geçmiştir [7].

Hancı tarafından 2017 yılında yapılan çalışmada, yazılım testi tasarım tekniklerinin matematiksel modellemesini ve bu modellemenin analizini vermiştir. Yazılım tasarım tekniklerinin ayrıntılı olarak tanıtıldığı bu çalışmada değerlendirme ölçütleri (her bir tekniğin kullanım sıklığı, hata bulma oranı, uygulanabilirlik) matematiksel modelde kullanılarak yazılım testi için en uygun olan teknikler tespit edilmiştir [8].

Kahveci tarafından 2017 yılında yapılan bu çalışmasında, bir bankanın yazılım kısmında çalışan proje yöneticileri için, fonksiyonel testleri için gerekli olacak test eforunu tahmin ederlerken kullanılacak bir yöntem geliştirmeyi amaçlamıştır. Bu çalışma sonucunda bir test analistine atanan her bir yazılım birimi için gerekli olan test çalıştırma eforunun test sürecinin ilk aşamalarında iken doğru bir şekilde tahmin edilebilmesi sağlanmıştır [9].

Baran, Akar ve Aslay, 2016 yılındaki çalışmalarında, önce geleneksel yazılım geliştirme yöntemlerindeki test sorunları özetleyerek, çevik yazılım geliştirme yöntemlerinde kullanılan en iyi test pratikleri araştırması sunmuşlar ve yazarların tecrübeleri ile birleştirilerek etkili test pratikleri önermişlerdir. Önerilen test pratiklerinin kullanımı ile daha kaliteli yazılım geliştirme süreçlerinin gerçekleştirileceğini öngörmüşlerdir [10].

Kayacan, 2017 yılındaki çalışmasında, yazılım proje başarısına etki eden kriterler dikkate alınarak bir karşılaştırma yapmıştır ve yazılım proje yönetimi metodolojisi seçimi konusunda önerilerde bulunmuştur. Bu karşılaştırmalarda, çevik yöntemler şelale yönteminden fazla çıktığı görülmüştür. Elde edilen sonucun, deneysel tecrübe ile karşılaştırılması maksadıyla yazılım sektöründen 97 çalışanın katılımıyla bir anket uygulanmıştır. Anket sonucunda, 1 kriterde şelale modeli tercih edilirken, 48 kriterde çevik yöntemler tercih edilmiştir. Elde edilen cevapların ağırlıkları kullanılarak metodolojiler

açısından genel bir karşılama oranı hesaplanmıştır. Metodoloji seçimini etkilediği değerlendirilen kriterlerin karşılama oranları çevik yöntemler için %67,4; şelale modeli için % 51,91 olarak elde edilmiştir [11].

Macit ve Tüzün 2015 yılındaki çalışmalarında, geleneksel şelale süreç modeli ile kenetlenme süreç modelini karşılaştırmışlardır. Özellikle öngörü sahibi olunan alanlarda şelale modeli ile başarılı sonuçlar alınmışken, bilgi almanın azaldığı ve proje takviminde sürenin uzadığı projelerde şelale modeli zorlanmıştır. Süreç seçimi için Stacey grafiğine benzer yaklaşımlar kullanmışlardır [12].

Gencer ve Kayacan tarafından 2017 yılında yapılan çalışmada, şelale modeli ile çevik yöntemler arasında karşılaştırma yapmış ve etkili olan faktörleri ele almışlardır ve yazılım proje yönetimi metodoloji seçimi konusunda önerilerde bulunulmuştur [13].

Yücalar ve Borandağ tarafından 2019 tarihinde yapılan çalışmada yazılım geliştirme süreçlerinde kaliteli ürün çıkarmak ve test süreçlerini iyileştirmek isteyen firmaların TMMi hakkında bilgilendirilmesi amaçlanmış ve bu firmaların test süreçlerinin olgunluk seviyelerini yükseltme noktasında nelere dikkate edilmesi gerektiği konusunu çalışmalarında ele almışlardır. [14].

Bose ve arkadaşları tarafından 2016 yılında yapılan çalışmada, TMMi sertifikası almak isteyen şirket içerisindeki organizasyonların dönüşümlerini ele alınmış ve mevcut test uygulamalarının ve istenen TMMi olgunluk seviyesine karşı ilk değerlendirmelerinin nasıl yapılacağı ile ilgili rehber oluşturmuşlardır. TMMi olgunluk seviyesinde istenen seviyeye ulaşmak için mevcut ve son durum farkını ele almak için örgütsel değişim yönetimine değinmişlerdir [15].

Araujo ve arkadaşları 2013 yılında yaptıkları çalışmada TMMi modelini referans alan küçük ve orta firmalar için yazılım olgunluk değerlendirmesi için bir model önererek bu model kapsamında bir değerlendirme anketi sunmuşlardır [16].

1.2. Tezin Amacı

Yazılım test süreçleri istenilen olgunluk seviyesinde olması için çeşitli olgunluk seviyesi ölçme ve iyileştirme modelleri bulunmaktadır. Bu modeller test süreçlerinde uygulanarak olgun bir test seviyesinde olan süreç yakalamaya çalışan şirketler, test süreç iyileştirme modellerini kullanarak daha kaliteli, hatası az bir şekilde ürün çıkarmaya odaklanmaya başlamıştır. Bu süreç modelleri uygulanarak belirli bir test olgunluk seviyesine ulaşarak daha etkin, verimli tüm organizasyon çalışanları tarafından benimsenen bir test süreci oluşturmak şirketler açısından vazgeçilmez bir hedef haline gelmiştir. Çalışmada test olgunluk seviyesi ölçümünde şirketlerin kendi öz süreçlerini kendilerinin değerlendirebilmesi için bir yöntem sunulmuştur. Bu yöntem ile şirketlerin, kendi iç test süreçlerinin olgunluk seviyesinin belirlenmesi ve değerlendirilmesi ile ilgili fikirler edinmeleri amaçlanmıştır. Böylece şirketlerin bu yöntemi kullanarak ekstra danışmanlık maliyetine girmeden kendi öz süreçlerini değerlendirebilecekleri ve değerlendirebilecekleri bir yöntem olması amaçlanmıştır. Çizelge 1.1'de test süreçlerinin olgunluk seviyesi ölçümü için farklı metodolojileri bulunmaktadır. Bu süreç modellerinden Test Olgunluk Model Entegrasyonu (TMMi) modeli kullanarak tez kapsamına ele alınan sistem üzerinde test süreç iyileştirme olarak 2nci ve sonrasında 3ncü olgunluk seviyesine getirilmesi hedeflenmiştir. Bu tezin amacı, önerilen yöntem ile olgunluk seviyesi belirleme ve değerlendirmede şirketlere kolaylık sağlaması ve geniş çaplı kabul edilir olan, dünyada bilinirliği fazla olan, Yetenek Olgunluk Model Entegrasyonu (CMMI) ile entegrasyonu ve uyumu bulunan, tümleşik kritik alanlar üzerine odaklanan ve dokümantasyon ve kaynaklar içeren TMMi model yaklaşımının bir örnek problem üzerinde uygulanabilirliğini göstermektir.

Çizelge 1.1. Test süreç iyileştirme modelleri [5]

Kısaltma	**Detay**	**Yazar/Organizasyon**
BTM	Beizer' in Test Modeli	1990 yılında Beizer tarafından hazırlanmıştır.
TOM	Test Organizasyonu Olgunluk Modeli	1990'ların sonunda İngiltere'deki Gerrad Consulting tarafından yazılmıştır.
TMAP	Test Yönetimi Yaklaşımı	Martin Paul, Rudd Teunissen ve Erik tarafından yazılmıştır 1995 yılında Veenendaal olabilir ve şu anda Capgemini'nin bir parçası olan Sogeti'ye aittir.
TSM	Test Edilebilirlik Destek Modeli	1996 yılında Dr. David Gelperin tarafından yazılmıştır.
TPI	Test Süreci İyileştirme Modeli	1996 yılında Tim Koomen ve Martin Pol tarafından yazılmıştır.
TMM	Test Olgunluğu Modeli	1996 yılında Illinois Institute of Technology'de Dr. Ilene Bernstein tarafından yazılmıştır.
TCMM	Teknik Yetenek Olgunluk Modeli	2001 yılında Torry Harris tarafından yazılmıştır.
TIM	Test Geliştirme Modeli	2002 yılında Thomas Ericson tarafından yazılmıştır.
TCI	Test Kabiliyetini Geliştirme	Atos Origin tarafından hazırlandı
TPA	Test Süreci Değerlendirmesi	CTG tarafından hazırlandı
TMAP Next	Yeni Nesil İçin Test Yönetimi Yaklaşımı	2005 yılında Tim Koomen, Michiel Vroon, Leo van der Aalst ve Bart Broekman tarafından yazılmıştır.
TMMi	Entegre Test Olgunluğu Modeli	2007 yılında TMMi Vakfı tarafından. 2010 yılında 4. ve 5. seviye çerçeveler yayınlanmıştır.
TPCR	Test Süreci ve Yetenek Değerlendirme	2011 yılında Barış Sarıgerioğlu tarafından hazırlanmıştır ve 2013 yılında revize edilmiştir.

1.3. Tez Çalışmasının Yapısı

Bu tez çalışmasının 1. bölümü olan Giriş bölümünde tezin amacı verilmiş ve sonrasında 2. bölümde yazılım geliştirme süreçleri ve 3.bölümde yazılım test süreçleri ele alınarak ayrı ayrı değinilmiştir. 4. bölümde test süreç iyileştirmeden bahsedilerek, test olgunluk modelleri verilmiştir. 5.bölümde CMMI modeli genel olarak verilmiş ve ardından 6. bölümde TMMi vakfı tarafından oluşturulmuş ve TMMi modeli için kılavuz kabul edilen TMMi modeli özetlenmiştir. 7.bölümde test olgunluk seviye belirleme için önerilen model verilmiş ve 8.bölümde TMMi modeli seviye 2 ve seviye 3 olgunluk seviyelerine geçiş süreci uygulamalı olarak proje üzerinde TMMi modeli uygulanarak bir sonuca ulaşılmıştır. Ayrıca uygulama bölümünde verilmiş olan test süreç iyileştirme çalışmasında bulunacak firmalara faydalı olacağı düşünülen doküman şablonları ve tablolar verilmiştir.

2. YAZILIM GELİŞTİRME SÜRECİ

Yazılım geliştirme süreci tanımlama, ayrıntılandırma, gerçekleştirme, değerlendirme, yayma ve yönetim aşamalarından oluşmaktadır. Bu aşamaların her biri kendi içerisinde alt bölümlere ayrılmıştır [17].

- Planlama
- Analiz
- Tasarım
- Gerçekleştirme
- Bakım
    - *Düzeltici bakım*
    - *Uyarlayıcı Bakım*
    - *İyileştirici/Yetkinleştirici Bakım*
- Yönetim

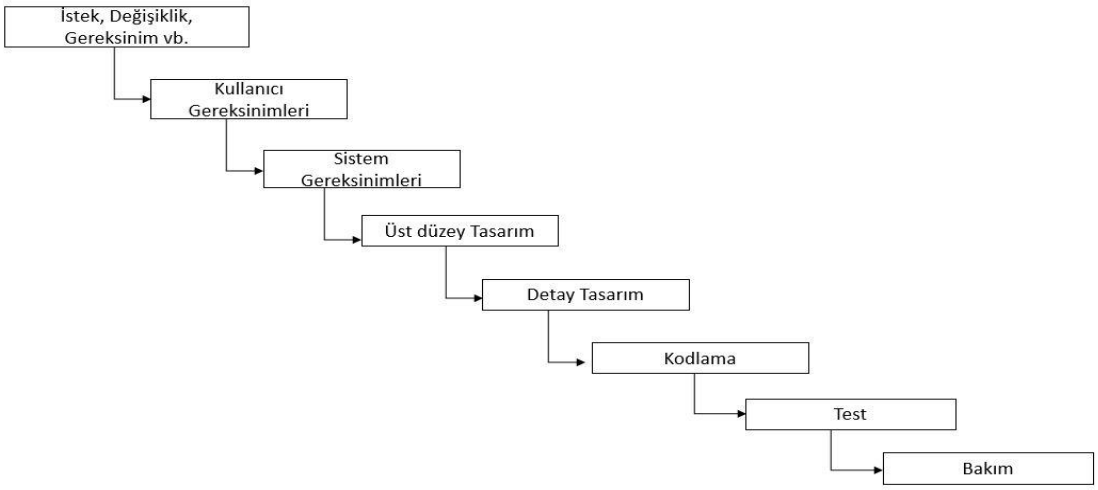
2.1.Yazılım Süreç Modelleri

Yazılım süreç modelleri, planlama, analiz, tasarım, gerçekleştirme, bakım ve yönetim adımlarını içerir. Aynı adımları içeren farklı modeller aşağıda verilmiştir.

2.1.1.Şelale (Waterfall) modeli

Öncelikle gereksinimlerin tanımlamasının yapıldığı bu modelde sistem ve yazılım tasarımları gerçekleştirilir. Sonrasında kodlama ve test etme süreçleri başlar. Testi başarılı geçtikten sonra bakım ve yönetim adımlarına geçiş yapılır. Şekil 2.1' de göründüğü gibi her bir aşama kendinden önceki aşamanın bitmesini beklemektedir. Buradaki en önemli sorun ise teste ayrılan zaman test etme adımından önceki aşamalarda gerçekleşen sorunlardan dolayı kısa olmasıdır. Bu modelde özellikle testten önce örneğin analiz aşamasında yakalanan hataların maliyeti çok yüksek olmaktadır. Hatalı durum için tekrarlı bir analiz,

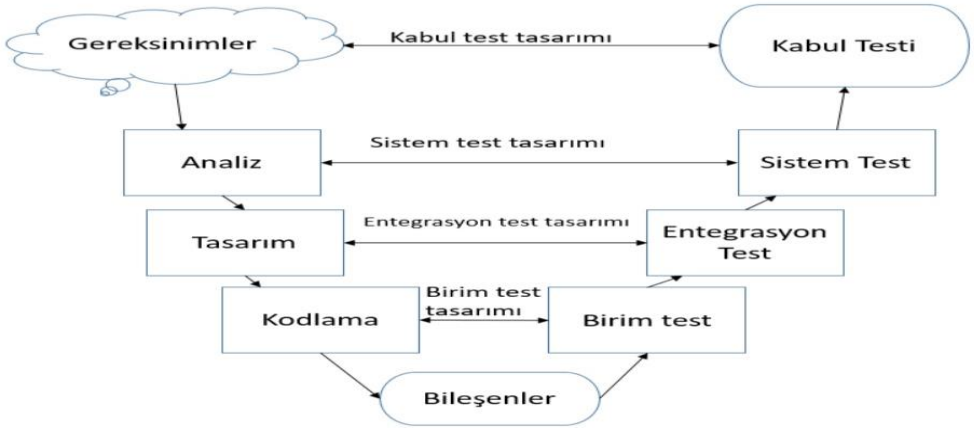
tasarım, kodlama ve test süreçlerinin işletilmesi gerekmekte ve plandan sapmalar oluşmaktadır [2].



Şekil 2.1. Şelale modeli

2.1.2. V-modeli

Bu modelde güvenilir ve doğru yazılımlar geliştirmek için bütün yazılım geliştirme süreç adımlarına karşılık gelen test aktiviteleri bulunur. Şekil 2.2' deki gibi şeklinden dolayı V-modeli ismini alan bu modele göre her bir yazılım sürecine farklı test yöntemleri Şekildeki gibi uygulanır, böylece çıkacak hataları daha önce bulunur. Yazılım geliştirme yaşam döngüsü boyunca bu model uygulanmış olur [17].



Şekil 2.2. V – modeli

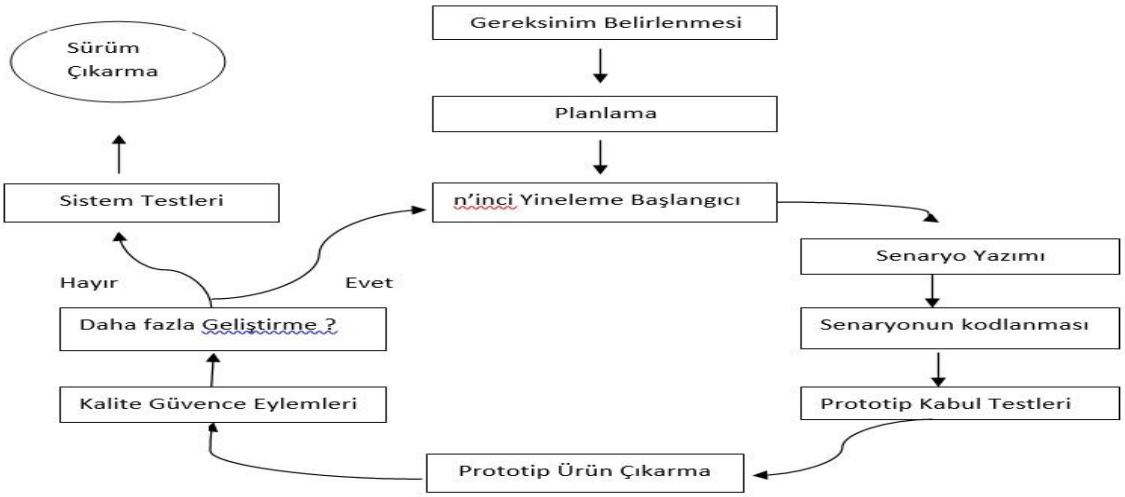
2.1.3. Prototip geliştirme modeli

Bu modelde gereksinimler toplanarak sürece başlanır ve sonrasında geliştiriciler ve kullanıcılar tarafından yazılımdan elde edilecek bütün çıktılara ve bu çıktılar için girdilerin nasıl sağlanacağına, nasıl korunacağına, hangi işlemlere uğrayacağına karar verilir. Tasarım yapıldıktan sonra kullanıcıya bir prototip üretilir. Kullanıcının kullanımında ve değerlendirilmesine sunulan prototip üzerinden değişiklikler yapılır. Böylece kullanıcının istediği ürüne yakın bir prototip üzerinde yazılım ile ilgili neler yapılacağına karar verilir [18].

2.1.4. Çevik (Agile) modeller

Çevik model ile yazılım geliştirmede, organize olan takımlar tarafından, uyumlu ve işbirlikçi bir anlayışla, dokümantasyonun en aza indirgendiği , pratik etkili bir yönetimle, zamanında, uygun maliyetli ve paydaşların değişen ihtiyaçlarına cevap verebilen, kaliteli çözümler üreten yinelemeli ve artımlı(evrimsel) bir yaklaşımdır [13]. Şekil 2.3' de çevik modele ait aşamalar verilmiştir.
Başlıca çevik modeller:

- Scrum
- Çevik Bilgi Yöntemi (Agile Data Method)
- Uç (sınırsal) Programlama (Extreme Programming-XP)
- Test güdümlü Geliştirme (Test-Driven Development)
- Özellik Güdümlü Geliştirme (Fature-Driven Programming)
- Çevik Birleştirilmiş Süreç (Agile Unifed Process)



Şekil 2.3. Çevik modeli [17]

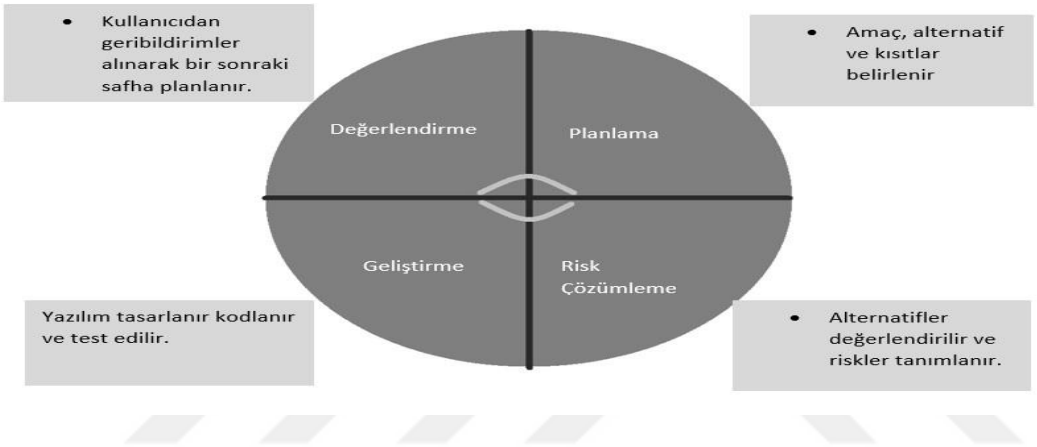
2001 yılında bu yöntemlere çevik yöntemler (agile methods) verilmiş ve bu oluşumun desteklenmesi için "Agile Alliance" organizasyonu kurulmuştur.
Çevik yöntemlerde uygulanan tekrarlamalar, önceki aşamalarda yapılan tecrübeler ve tespit edilen problemler ile şekillenir. Yapılacak işler önceliklendirilerek işin nasıl yapılacağını, mevcut kaynaklar ve kısıtlar ile projede bulunan takım üyeleri tarafından belirlenir. Bu nedenler takım içi uyumun önemli olduğu bu modelde çevik yöntemlerin baz alındığı temel durumlar, önceliklendirme, deneysellik, kendi kendini organize edebilme, zaman yönetimi ve iş birliği şeklinde sıralanabilir [13].

2.1.5. Evrimsel geliştirme modelleri

İlk gerçekleştirimi hızlı yapıp onun üzerinde kullanıcının tepkilerini alıp iyileştirmeyi öngörerek yazılım geliştirilmesi hedeflenen bu modelde müşteriyle çalışarak ana hatları belli olan son yazılımı geliştirmektir. Bu yaklaşım yazılımın gereksinimlerinin daha iyi anlaşılmasını hedeflemektedir. Atılabilir prototiplemede amaç yazılım gereksinimlerini keşfetmektir. Bu nedenle ilk tespit edilen gereksinimlerden bir prototip yazılım geliştirilerek müşteriye gösterilir. Müşterinin istediği ürün anlaşılmaya çalışılır. Bu nedenle ilk geliştirilen ürün örnek bir üründür ve daha sonra geliştirilecek olan yazılımlar bu ilk yazılımı temel alarak geliştirilir. Kısa sürede yazılımın geliştirilmesinin istendiği küçük ve orta büyüklükteki projelerde bu yaklaşım rahatlıkla kullanılabilir [17].

2.1.6. Sarmal model

Her bir artırımda güvenli ve hızlı yapılması amaçlanan bu modelde, her bir iterasyonda yazılıma ait yeni sürüm ortaya çıkar. İlk iterasyonda oluşan sürüm, ilk örnek gibi düşünülür. İlk örnek olarak düşünülen üründen tek farkı uygulama ortamında kullanılan gerçek yazılım olmasıdır. Sarmalın her döngüsü kendi içinde yapılması gereken Şekil 2.4' deki gibi etkinliklere ayrılır. Bu etkinlikler projelerin boyutuna ve karmaşık yapısına göre güncellenebilir [18].



Şekil 2.4. Sarmal modeli [17]

Sarmalın her bir döngüsü sırayla tekrarlanan dört etkinlikten oluşur:

- Planlama; yapılacakların eldeki kaynaklar ve zaman çizelgesi gibi konular göz önünde tutularak planlanması.
- Risk çözümleme; teknik ve yönetsel risklerin çözümlenmesi.
- Geliştirme; yazılımın gerçekleştirimi, çözümleme, tasarım, kodlama.
- Değerlendirme; kullanıcı geribildiriminin alınması, yapılanın geçerliliğinin ve gerektiği gibi çalıştığının ortaya konması [17].

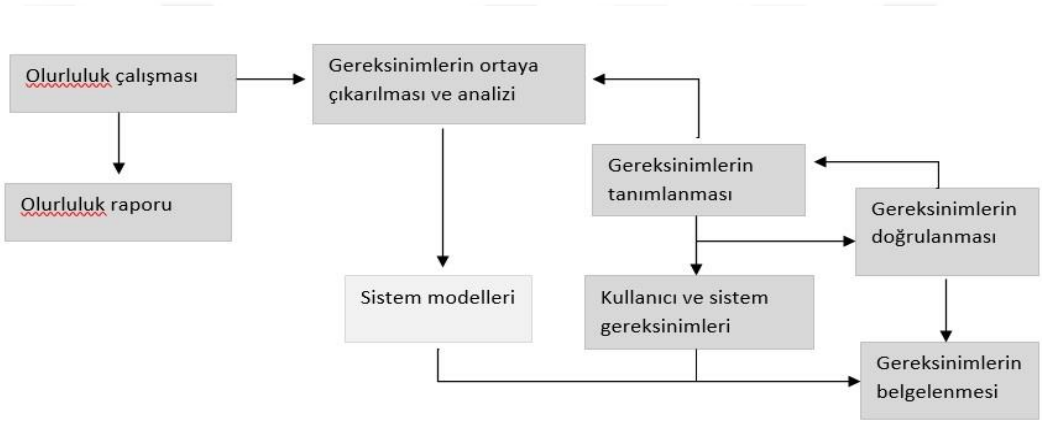
2.1.7. İteratif model

Şelale modelinin alternatifi olarak düşünülen bu modelde belirli bir süresi bulunan ve bu süre içerisinde yazılım bölünerek belirlenen süre içerisinde bitirmek amaçlanır. Her bir adımın süresi belirlendiği için olası gecikmeler sadece ilgili iterasyonu etkiler. Bundan

dolayı yönetilebilirlik diğer modellere kıyasla daha iyidir. Ayrıca çalışanlar her yeni iteratifte geçildiğinde uygulama alanına ilişkin daha çok deneyim kazanılmış olur [17].

2.2. Yazılım Gereksinim Belirtimleri

Yazılım gereksinimi geliştirilen yazılımdan ne beklendiğinin ifade edilmesidir. Amacına uygun yazılımların geliştirilmesi için beklenenlerin yani yazılım gereksinimlerinin çok iyi tespit edilmesi gereklidir. Bu amaçla yazılım geliştirmenin ilk adımında yazılım gereksinim analizleri Şekil 2.5' de görülen süreç ile gerçekleştirilir. Bu süreç aynı zamanda gereksinim mühendisliği süreci olarak da bilinir [17].



Şekil 2.5. Gereksinim analiz süreci

Bu süreç temel olarak aşağıdaki adımları kapsar:

- Olurluluk çalışması ve raporlanması
- Gereksinimlerin ortaya çıkarılması ve analizi
- Gereksinimlerin tanımlanması
- Gereksinimlerin doğrulanması

Yazılım gereksinimlerinin ortaya çıkarılması başarılı yazılımlar geliştirilmesi açısından çok önemlidir. Çünkü doğru anlaşılmış bir ihtiyaç doğru yazılımın geliştirilmesi için atılan en önemli adımdır. Bu evre içerisinde daha önemli olan adım ise belirlenen gereksinimleri belgelendirilmeden önce gözden geçirmeler ile doğrulanmasıdır. Gözden geçirmelerde gereksinimler doğruluk, açıklık, anlaşılırlık, yapılabilirlik, tutarlılık, izlenebilirlik,

uyumluluk, test edilebilirlik gibi bir dizi kriter açısından gözden geçirilir. Amaç ihtiyacın doğru tespit edilmesi, tespitin yapılabilir olması ve yapıldıktan sonra da test ederek yapılabildiğinin gösterilmesidir. Bu işlem yazılım geliştirmenin en erken safhalarında hataların yakalanması ve düzeltilmesi açısından oldukça önemlidir [17].

2.3. Yazılım Tasarımı ve Geliştirilmesi

Tasarım, herhangi bir mühendislik alanında geliştirilecek olan ürünün ilk modelinin ve gösteriminin ortaya çıkarılmasıdır. Bu açıdan tasarımların gerçekleştirilmesinde deneyim ve bilgi birikimi önemli yer tutar. Yazılım geliştirme sürecinde tasarım ve geliştirme tespit edilen yazılım isterlerinin çalışılabilir bir yapıya dönüştürülmesi evresidir. Yazılım tasarım sürecinde aşağıdaki tasarımlar gerçekleştirilir:

- Bileşen tasarımı
- Algoritma tasarımı
- Mimari tasarım
- Veri yapısı tasarımı
- Ara yüz tasarımı

Bu tasarımların ortaya çıkmasından sonra hedeflenen yazılımın kodlanmasına başlanır. Tasarım ile geliştirme faaliyetleri birbirine çok yakındır ve seçilen yazılım geliştirme süreç modeline göre iç içe gerçekleştirilebilen eylemlerdir [17].

2.4. Yazılım ve Kalite

Bir yazılımın kalitesi, kullanım amaçlarına göre açık ve anlaşılır şekilde tanımlanmış fonksiyonel gereksinimlere uygunluk, müşteri isteklerini karşılayabilme, açık ve net şekilde dokümante edilmiş yazılım geliştirme standartlarına bağlılık, güvenilirlik ve yazılım ürünün teslim sonrası destek sağlama olarak tanımlanabilir. Gereksinimler ya da müşteri istekleri kalitenin ölçülmesinde referans alınabilirler. Bunların dikkate alınmaması yazılım işlevselliğinden uzaklaşmak anlamına gelir. Belirlenmiş standartlarda geliştirilen bir yazılım ürünü belirli bir teknik kalite seviyesindedir. Geliştirme kriterlerinin uzağında olması

durumunda kalitesiz ve fonksiyonelliği içermeyen bir yazılım durumunda kalmış olur. Yazılımda bulunması gereken sağlamlık, anlaşılabilirlik, bakım kolaylığı, doğruluk, modülerlik, bakım kolaylığı gibi özelliklerin eksikliği, yazılımın kalitesinde istenilen seviyede olmaması anlamına gelir [22].

Basit anlamda yazılımda kalite hataların en aza indirgenmiş ya da olmaması olarak açıklanabilir. gereksinimlere uyumluluk önemlidir çünkü yazılımda fonksiyonel hata fazla ise gereksinimlerde istenen işlevlere uyuşmaz. Bu tanım iki kavramı içerir; hata oranı, yazılımı geliştirilen koddaki satır başına düşen hata sayısı olarak söylenebilir. Güvenilirlik ise belirlenen bir (n) zamandaki başarısızlık sayısıdır [23].

Yazılımın kalitesi için yapılacak doğrulama ve geçerleme geliştirilen yazılımın doğru çalıştığının ve müşterinin beklentilerini karşıladığını göstermeyi amaçlar. Bu amaçla yazılım gözden geçirmesi ve diğeri yazılım testleri kullanılan araçlardır. Gözden geçirmeler ile hataların geliştirim sürecinin erken safhalarında yakalanması ve bu hataların düzeltilmesi sağlanır. Bu gözden geçirmeler:

- *Yönetim Gözden Geçirmeleri:* Plan ve proje takviminin izlenerek belirlenen zamandan saptanması, projede belirlenen hedeflere ulaşılabilmesi için uygulanan yaklaşımların değerlendirilmesi amacıyla yönetim tarafından ya da yönetim adına yapılan değerlendirmelerdir.

- *Teknik Gözden Geçirme:* Kalifiye (nitelikli, ehil) bir ekip tarafından bir iş ürününün hedeflenen amaca uygunluğunun saptanması, gözden geçirilen iş ürününün kendi belirtimlerinden ve standartlardan sapmalarının tanımlanması amacıyla yapılan sistematik değerlendirmelerdir.

- *İnceleme (Inspection):* Bir iş ürünündeki hataları ve standartlardan sapmaları içeren anormalliklerin belirlenip tanımlanması için inceleme teknikleri ile ilgili eğitimli veya tecrübeli tarafsız kişilerin veya denk kişilerin katılımıyla gerçekleştirilen görsel sorgulamadır.

- *Denetleme (Audit):* Bir iş ürününün veya bir sürecin, belirtimler, standartlar, sözleşme veya diğer kriterler bakımından uyumunun değerlendirilmesi için yapılan sistematik bağımsız değerlendirmedir.

- *Üzerinden Geçişler (WalkThrough):* Bir yazılımın, kodlayıcısı tarafından diğer yazılım geliştirme ekip üyelerine anlatılarak yazılım ürününün iyileştirilmesine yönelik görüşlerin alınması ve projenin kodlama standardına uyulmadığı noktaların veya olası hataların tespit edilmesidir [17].

Yazılım testleri ise yazılım içerisindeki hataların keşfedilmesine ve bunların düzeltilmesinin sağlanmasına odaklanır.

2.5. Yazılım Süreç Modellerinde Yazılım Testleri

Yazılım insanlar tarafından geliştirilen bilgisayar kodlarıdır. Bu nedenden dolayı yazılımlarda hataların ortaya çıkması kaçınılmazdır. Önemli olan ortaya çıkan veya çıkabilecek olan bu hataların zamanında belirlenmesi ve düzeltilmesidir [22].

Yazılım süreç modelleri yazılı geliştirmede var olan hataları öngörerek yazılım geliştirmeyi en etkin ve doğru şekilde gerçekleştirilmesini sağlamayı amaçlamaktadır. Bu modeller doğru ve etkin uygulanarak yazılım projeleri zamanında ve maliyeti aşmadan tamamlanabilir. Ancak geliştirilen yazılımların gereksinimleri karşılayıp karşılamadığı ve hataları en aza indirgenmiş bir yazılım geliştirilmesi bu süreç modellerinde uygulanan test süreçleri ile sağlanır. Bu açıdan yazılım testleri yazılım geliştirme süreci içerisinde en önemli aşamalardan biridir. Yazılım test süreçleri detaylı olarak sonraki bölümlerde verilmiştir.

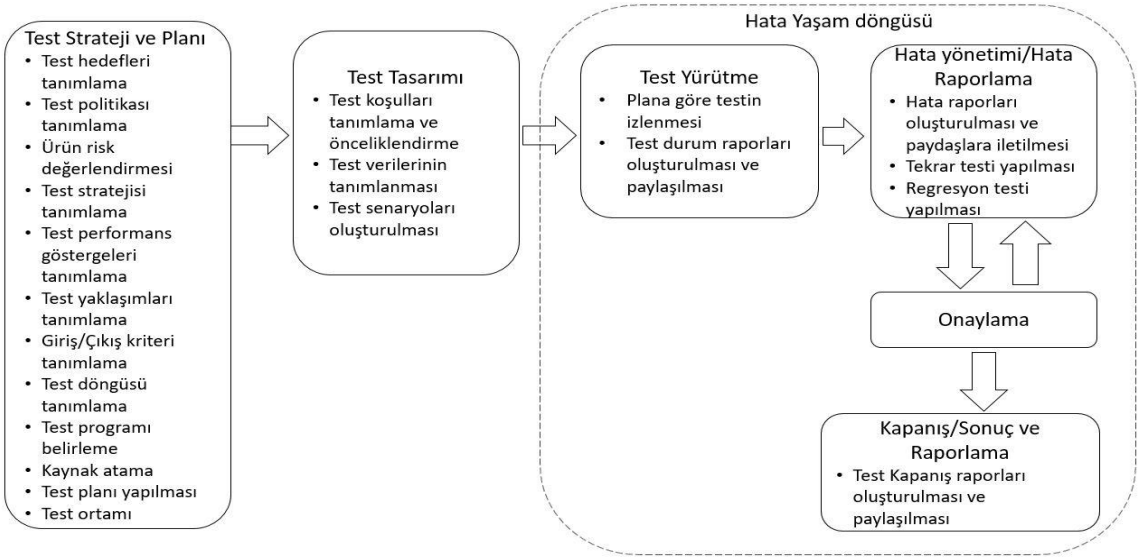
3. YAZILIM TEST SÜRECİ

3.1.Yazılım Test Süreci

Test, kullanıcı gereksinimlerinin karşılanıp karşılanmadığı ve sistemin işlevselliğinin kontrol edildiği aktivitelerdir. Hataların bulunması, kalite ile ilgili güvenilirlik, karar verme için bilgi sağlama ve hataları önleme testin hedefleri olarak sıralanabilir. Testlerde en belirgin aşama testin yürütüldüğü aşamadır. Fakat test koşumu öncesinde ve sonrasında gerçekleştirilen işlemlerde bulunmaktadır. Bu işlemler arasında planlama, test koşullarının seçilmesi, test senaryolarının tasarlanması ve koşturulması, sonuçların kontrol edilmesi, çıkış kriterlerinin değerlendirilmesi ve raporlanması, testi sonuçlandırma ve kapatma yer alır.

Yazılım test süreci planlanan ve gerçekleştirilip sonuçları kaydedilerek belgelendirilen aktivitelerden oluşur. Şekil 3.1' de yazılım test süreci görülmektedir. Bu süreç geliştirilen yazılımdaki hataların varlığına odaklanır. [17].

Bir yazılım test sürecinde olması gereken aşamalar ve aşamalarda yapılması gerekenler aşağıdaki Şekil 3.1' de verilmeye çalışılmıştır. Test strateji ve plan adımında belirtilenler yapılarak, test tasarımı adımına geçilir. Test tasarımı adımında test durumları oluşturulur ve testler yürütülmesi için bir sonraki adıma geçilir. Bu adımla birlikte açılacak hatalar ve hataların çözülmesi sonrasında tekrar testleri ya da sistem de etkilenen alanların olup olmadığının testi için regresyon testi gerçekleştirilir. Hataların düzeltildiğin onaylama aşamasından sonra test durum raporları, test kapanış raporları oluşturulur ve paylaşılır. En genel anlamda bir test süreci bu şekilde ilerleyebilir.



Şekil 3.1. Yazılım test süreci

3.1.1. Test planlama

Test planlaması testin amacının belirlenmesi test aktivitelerinin amaca uygun yapıldığını gösteren yazılım test sürecinin ilk adımıdır. Test kontrolü planlamaya göre gerçek süreçten sapmalarının olup olmadığının kontrolünün yapıldığının göstergesidir. Test planları proje kapsamındaki test stratejisinin tanımlanması, kaynak planlamasının yapılması, sorumlulukların, risklerin ve önceliklerin açıkça belirtildiği dokümanlardır. Test planı oluşturulurken basit, güncel, anlaşılır, açık ve kabul edilebilir olmalıdır [21]. İyi bir test planı geliştirmek için aşağıdaki adımlar takip edilmelidir:

- Test amaçları tanımlanmalıdır.
- Test stratejisi (yaklaşımı) belirlenmelidir.
- Test ortamı tanımlanmalıdır.
- Test belirtimlerinin nasıl geliştirileceği belirlenmelidir.
- Test takvimi belirlenmelidir.
- Test planı gözden geçirilmeli ve onaylanmalıdır [17].

3.1.2. Test analizi ve tasarımı

Test analizi ve tasarımı süreci planlama süreci tamamlandıktan sonra başlar. Test tasarımı sırasında gerekli bilgiler toplanır ve bunlara bağlı kalarak tasarıma geçilir. Test tasarımda, hatalı durumu bulacak şartlar ve bu şartlar altında yapılması gerekenler ve daha sonra istenilen durumun ne olduğunun belirtilmesi en önemli noktadır [21]. Test tasarımı sırasında, test koşulları, üst düzey test durumları ve diğer test yazılımları detaylandırılır. Bu nedenle test analizi, test tasarımı aşamasında "ne test edilmeli" ve "nasıl test edilir" sorularına cevap verilir. Test tasarım sürecinde aşağıdaki işlemler gerçekleştirilir:

- Test senaryolarının tasarlanması ve önceliklendirilmesi
- Test koşullarını ve test durumlarını desteklemek için gerekli test verilerinin belirlenmesi
- Test ortamını tasarlamak ve gerekli altyapı ve araçların belirlenmesi
- Testin temeli, test koşulları, test durumları ve test prosedürleri arasında izlenebilirlik oluşturulması

Test analizinde olduğu gibi test tasarımı da test sırasında benzer tipteki hataların tanımlanmasına neden olması önemli bir katkı olarak görülür.

3.1.3.Test uygulaması ve yürütülmesi

Test uygulaması sırasında, test uygulamalarını test prosedürlerine sıralamak da dâhil olmak üzere test uygulaması için gerekli test yazılımı oluşturulur veya tamamlanır. Test tasarımı sırasında "nasıl test edilir" sorusu cevap bulurken, test uygulaması sırasında "testleri yapmak için gerekli olanlar var mı" sorusu cevaplanır. Test uygulaması sırasında gerçekleştirilen temel işlevler aşağıdaki gibidir:

- Test prosedürlerini geliştirme ve önceliklendirme ve potansiyel olarak otomatikleştirilmiş test komut dosyaları oluşturma

- Test prosedürlerinden ya da varsa otomatikleştirilmiş test komut dosyalarından test paketleri oluşturma

- Test süitlerini bir test yürütme programı dahilinde, verimli bir test yürütmesiyle sonuçlanacak şekilde düzenlemek

- Test ortamının oluşturulması (potansiyel olarak test betiklerinin yazılması, servis sanallaştırılması, simülatörler, ve diğer alt yapı öğeleri) ve ihtiyaç duyulan her şeyin doğru şekilde ayarlandığını doğrulamak

- Çift yönlü izlenebilirliğin doğrulanması ve güncellenmesi

Test yürüme sırasında gerçekleştirilen temel aktiviteler aşağıdaki gibidir:

- Gerçekleştirilen testlerin sürümlerin kaydedilmesi
- Testleri manuel ya da otomatik şekilde yürütülmesi
- Gerçek sonuçla beklenen sonuçların karşılaştırılması
- Testlerde karşılaşılan hatalar için testlerin tekrarlanması (düzeltilmiş bir hatanın testi, onay testi ya da regresyon testi yapılması)
- Karşılaşılan hataların analizinin yapılması
- Karşılaşılan hataların bildirilmesi
- İzlenebilirliğin doğrulanması ve güncellenmesi
- Test yürütme sonucunun raporlanması, kaydedilmesi

Test koşturma yani testlerin gerçekleştirilmesi birim testler ile başlar ve yazılımın kodlanması aşamasında gerçekleştirilen birim testlerin sorumluluğu yazılımcılara aittir. Ancak testlerin denetlenmesi, birim ve modül testlerinde kullanılacak olan test durumlarının oluşturulma stratejisinin belirlenmesi testçilere aittir. Testçiler tarafından icra edilecek olan entegrasyon testlerinin tamamlanması ile sistem gereksinimlerinin doğrulanması amacıyla sistem testleri gerçekleştirilir. Sistem testlerinde işlevsellik, performans ve sistem kullanılabilirliği gereksinimleri test edilir. Test koşturmada son adım kullanıcı kabul testleridir. Bu testlerde sistemin kullanıcı gereksinimlerini karşıladığı doğrulanır.

Test durumu statüleri

Yazılım projelerinde, test sürecindeki test eylemlerinin izlenmesi ve raporlanması amacıyla test durumlarına ve senaryolarına çeşitli statüler özellik olarak atanabilir.

Çizelge 3.1. Test durum statüleri

Durum	**Açıklaması**
Passed (Geçti)	Koşturulan ve başarı ile sonuçlanan test durumlarını veya senaryolarını belirtmek için kullanılan durumdur.
Blocked (Bloklandı)	Testin başladığını ancak başka bir sebepten dolayı bloklandığını göstermek için kullanılan durumdur.
Retest (Tekrar test)	Tekrar gerçekleştirilecek olan test durumudur.
Failed (Hatalı)	Koşturulan ve başarısızlıkla sonuçlanan test durumlarını veya senaryolarını belirtmek için kullanılan durumdur.
Incomplete (Tamamlanamadı)	Testine başlanan ancak tamamlanmayan test durumudur.
No Run (Koşulmadı)	Henüz koşulmayan test durumudur.

3.1.4.Hata yönetimi ve hata raporlama

Gereksinimlerde belirtilen beklenen sonuç ile testler sırasında gerçekleşen sonucun uyuşmamasına hata denir. Testler sırasında karşılaşılan bu hatalar yazılımcılara bildirilir ve raporlanır. Hatalar düzeltildikten sonra onay testi ve sistemde etkilenip etkilenmeyen yerler olup olmadığının kontrolü için regresyon testleri gerçekleştirilir. Hata raporlama test araçları ya da doküman oluşturularak raporlanabilir ve kaydedilebilir. Hataların tamamının giderildiği ya da bitiş kriterlerinin karşılanması durumunda testler sonlandırılır.

3.1.5. Testin tamamlanması ve sonuç raporlama

Testin tamamlanması, deneyim, test yazılımı ve diğer ilgili bilgileri birleştirmek için tamamlanan test sürecinden veri alır. Test sürecindeki aktiviteler yazılım sisteminin tesliminin yapılması, projenin sonlanması gibi proje kilometre taşlarından oluşur.
Test tamamlama ana faaliyetleri aşağıdaki gibidir:

- Bulunan tüm hataların çözülüp, tekrar testinden başarılı gerçekleşip gerçekleşmediğini kontrol etme, test yürütme sırasında çözülmeyen hatalar için açılan değişiklik talepleri
- Test yazılımının bakım için hazır hale getirilmesi ve bakım ekiplerine, diğer proje ekiplerine ve diğer paydaşlara teslim etmek
- Paydaşlara iletilmek üzere bir test özet raporu oluşturulması
- Test ortamını, test verilerini, altyapısını daha sonra kullanmak üzere diğer test yazılımlarını sonuçlandırmak ve arşivlemek
- Daha sonra gerçekleştirilecek projeler için gereken değişiklikleri belirlemek için test faaliyetlerinden öğrenilmiş derslerin analiz edilmesi
- Test süreci olgunluğunu artırmak için toplanan bilgilerin kullanılması
- Testler yapıldıktan sonra test verileri raporlanması, analiz edilmesi ve değerlendirilmesi
- Değerlendirme de test planlarında belirtilen test geçme/kalma kriterleri

3.2.Test Yönetimi

Büyük ve karmaşık yazılımların gereği gibi test edilmesi oldukça zor ve zahmetli bir iştir. Testlerin etkin şekilde gerçekleştirilmesi farklı grupların iyi bir şekilde organize edilmesine, işlerin iyi planlanmasına, sorumlulukların ve ilişkilerin net tanımlanmasına bağlıdır. Test yönetimi kavramı test sürecinin ve bu süreç içerisinde yer alan ekibin, yönetimini kapsamaktadır [17].

3.2.1.Test aktörleri

Test liderinin ve test uzmanının yaptığı görevler, projenin ve ürünün büyüklüğüne, karmaşıklığına ve o rolü üstlenen kişilere veya kuruluşlara göre değişiklik gösterir. Test liderine test yöneticisi ya da test koordinatörü, test uzmanına da test mühendisi adı verilebilir. Genel olarak test lideri görevleri şunları içerir:

- Test stratejisini belirleme ve koordine etme ve diğer paydaşlarla planlama
- Kuruluşun test politikasının yazılması veya gözden geçirilmesi

- Ekip içi sorumlulukları belirleme
- Test hedeflerini ve riskleri göz önünde bulundurarak testleri planlama
- Testlerin analizini, hazırlığını ve yürütülmesini başlatma, test sonuçlarını izleme ve çıkış kriterlerini kontrol etme
- Test yazılımlarının izlenebilirliği için yapılandırma yönetimini hazırlama
- Test senaryolarının hangi dereceye kadar ve otomasyona geçiş planlamalarına karar verme
- Test raporları yazma
- Test yöneticisi deneyimiyle testlerde kullanılacak olan yazılımların seçimine yardımcı olma
- Proje içerisinde test faaliyetlerinde genel kabul görmüş test standartlarının, metotlarının ve kriterlerinin uygulanmasını sağlama [21].

Test uzmanının görevleri ise aşağıdaki gibi sıralanabilir:

- Test planlarını gözden geçirme ve test planı oluşturulma sürecinde bulunarak katkı sağlaması
- Test ortamının hazır durumda bulunması (Sistem yöneticisi ya da ağ yönetimi ile koordineli çalışma)
- Test verilerinin hazırlanması
- Test seviyelerinde (Birim, Sistem, Fonksiyonel vb.) testleri yürütme ve raporlama
- Testleri otomasyona geçirme
- Testleri planlandığı şekilde gerçekleştirme
- Gereken durumlarda test yazılımlarının alınmasında görüş belirtmek ve gerekli incelemeleri yapma
- Test durumlarını tasarlamak ve belgelendirmek
- Testleri gerçekleştirmek
- Test sonuçlarını ve tespit edilen hataları raporlamak
- Test tasarımı ve test sonuç raporlarının gözden geçirilmesinde katkıda bulunmak

- Gözden geçirme faaliyetleri sonucunda gereken durumlarda test tasarımındaki veya test sonuç raporlarındaki güncellemeleri gerçekleştirmek
- Testlerin yinelenmesinde görev almaktır. [7].

3.3.Yazılım Test Düzeyleri

Testin hangi derinlikte yapılacağı ile ilgili olan test seviyeleri aşağıdaki gibidir:

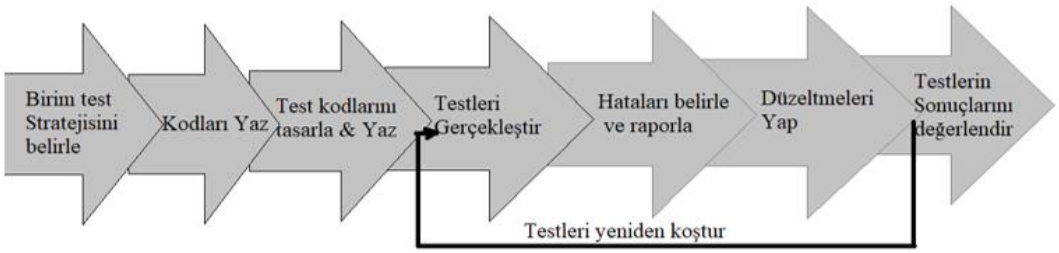
- Bileşen (Birim) Testleri
- Entegrasyon Testleri
- Sistem Testleri
- Kabul Testleri

3.3.1.Birim testler

Birim testleri, uygulamada olması muhtemel hataların daha önceden saptanıp çözülmesi hedefiyle test ekibine iletilmeden önce kod kapsamında uygulamayı test eden geliştiriciler tarafından yapılan testlerdir [24].

Birim testlerin gerçekleştirilmesi

Geliştirilen en küçük kod parçalarının (metot, prosedür, fonksiyon, sınıf vb.) doğru şekilde çalıp çalışmadığının kontrolü amaçlanır. Şekil 3.2’ de olduğu gibi gerçekleştirilen bu testler yapılırken yazılımcı test edilecek olan birimi diğer birimlerin doğru olduğunu düşünerek testi gerçekleştirir. Bu yüzden diğer birimlerle olan alışverişler stub (Koçan) ya da driver kullanır. Testlere başlamadan önce test senaryoları oluşturularak test edilecek birim çalıştırılır ve elde edilen sonuçlar beklenen sonuçlar ile karşılaştırılarak testlerin değerlendirmesi yapılır. Eğer test başarısızlıkla sonuçlanmış ise birimin kodu tekrar gözden geçirilir, gerekli düzeltmeler gerçekleştirilir ve tekrar test edilir [17].



Şekil 3.2. Birim testi süreci [17].

3.3.2.Entegrasyon testleri

Birim testlerden sonra gerçekleştirilen bu testler, farklı birimlerin ya da sistemlerin birlikte çalışırken karşılaşılacak sorunların test edilmesidir. Test durumları yazılım gereksinimleri ile ara yüz gereksinimlerinden çıkartılan bu testlerde kara kutu test yöntemi kullanılır. Testlerde olumlu ve olumsuz test durumları uygun parametreler kullanılarak gerçekleştirilir. Sonuçlar beklenen değerlerle karşılaştırılarak sistemin kendisinden beklenen davranışları gerçekleştirdiği doğrulanır [7].

3.3.3.Sistem testleri

Sistem testi, test planında açıkça belirtilerek, sistemin fonksiyonel ve fonksiyonel olmayan gereksinimlerini ve kalitesini sorgulayan testlerdir. Genel olarak test uzmanlarının yaptığı bu testlerde gereksinimlerin tamamlanmış olması aranır. Fonksiyonel gereksinimlerle ilgili sistem testinde kara kutu test teknikleri kullanılır. Fonksiyonel olmayan gereksinimlerin karşılandığı bu test seviyesinde fonksiyonel olmayan testler gerçekleştirilir [21]. Bu testler:

- *Stres Testleri:* Sisteme girdi oranının sistem tasarım oranını aşması sağlanarak sistemin baskı altında tutulması sağlanarak sistemin davranışı test edilir. Bu testlerde sistemin ciddi hatalar vermemesi beklenir.

- *Performans Testleri:* Sistem çıktılarının belirlenen kısıtlar ve belirli yük altında sistemde çıkacak darboğazları ortaya çıkarmak için yapılır. Ortamın stabil olması performans testleri için önemlidir ve asıl amaç sisteme yapılan girdilerden alınan çıktıların istenilen ya da olması gereken sonuçla uygunluğun test edilmesidir.

- *Konfigürasyon ve Uyumluluk Testleri:* Farklı platformlarda(Chrome, IE, Firefox vb.) ve donanımlarda sistemin nasıl davrandığının değerlendirilmesi için gerçekleştirilen testlerdir.
- *Güvenlik Testleri:* Yazılımın iç ya da dış ortam ya da çevre bazlı yetkisiz erişimlere, iyi olmayan amaçlar ile ilgili kullanımlara karşı sistemin davranışı ve güvenliğinin yeterliliğini değerlendirmek amacıyla yazılımdaki açıkları tanımlamak ve gidermek için gerçekleştirilen testlerdir. Bu testten geçebilmek için yazılımın tasarım ve kodlama aşamasında risk noktaları oluşturulmalı ve bu noktalara karşı yazılımsal önlemler alınmalıdır.
- *Kullanılabilirlik Testleri:* Kullanıcıların dahil edilerek kullanılabilirlik test araçları kullanılarak kullanıcı ile sistem arasındaki etkileşimi değerlendirmek için yapılan testlerdir.
- *Geri Alma Testleri:* Yazılımda çıkacak bir hata durumunda sistemin otomatik veya manuel bir şekilde mevcut duruma getirilmesini değerlendirmek için yapılan testlerdir.
- *Kullanıcı ara yüzü Testleri:* Kullanıcının ve yazılımın grafik gösterimi arasındaki etkileşimi değerlendirmek, kullanıcının sisteme vereceği girdilerin sistem tarafından nasıl işleneceğini değerlendirmek için gerçekleştirilen testlerdir.
- *Duman Testleri:* Geliştirilen yazılıma detaylı bir şekilde bakılmadan sistemin temel fonksiyonları yerine getirip getirmediğine bakıldığı bu test ile oluşacak ana yazılımın, test edilebilirliği kontrol edilir. Bu amaçla yazılımın temel sürecinden birkaç senaryo seçilir ve koşturulan testler sonrasında testin başarı oranına göre yazılım testine başlanabilir ya da reddedilebilir.
- *Keşif Testleri:* Keşif testleri aynı anda öğrenmenin, test tasarımının ve test koşturulmasının gerçekleştirilmesidir. Yapısal bir yaklaşım sergilenmez. O anki duruma göre testin bir sonraki adımı gerçekleştirilir. Her şey anlık ve geçici olarak gerçekleştirilir. Bu yaklaşım ilk olarak Cem Kaner tarafından "Testing Computer Software" kitabında verilmiştir.
- *Yükleme Testleri:* Yazılıma çökene kadar yüklenilerek yazılımın sınırlarının ölçüldüğü veya maksimum yük seviyelerinin belirlendiği testlerdir.
- *Uygunluk Testleri:* Yazılımın kendi gereksinimlerine uygun geliştirilip geliştirilmediğini değerlendirmek için yapılan testlerdir. Geliştirilen yazılımın

taşınabilirlik, birlikte çalışılabilirlik gibi kendisiyle ilgili gereksinimleri sağladığının gösterilmesi bu testlerin amacıdır.

- *Yineleme Testleri***:** yazılım üzerindeki yeni bir işlevsel birim geliştirildikten veya bir düzeltici faaliyet gerçekleştirildikten sonra icra edilen testlere denir. Amaç yapılan geliştirmenin veya düzeltme faaliyetinin yazılımın diğer bölümlerini olumsuz yönde etkilemediğinin gösterilmesidir. Testlerin otomatik olarak gerçekleştirilmesi zamanın ve kullanılan kaynakların azaltılmasını sağlayabilir.

3.3.4. Kabul testleri

Proje kabul testleri müşteri gereksinimlerinin doğrulanması için müşteri katılımı ile kullanıcı gereksinimleri için test durumları oluşturulduktan sonra gerçekleştirilen bu test seviyesinde aşağıdaki testler bulunmaktadır:

- *Kullanıcı Kabul Testleri:* Sistemin son kullanıcılarından ya da müşterilerden gelen gereksinimlerin karşılanıp karşılanmadığı test edilir.
- *Uyumluluk testleri:* Geliştirilen ürününün kullanıldığı ülkenin standart kurallarına uyumluluk ve diğer kişisel haklarının ihlalinin olacağı durumlar test edilir. Chrome, IE, Firefox gibi tarayıcılarda geliştirilen web uygulamalarında oluşabilecek fonksiyonel, performans ve görsel gibi durumların test edilmesi yer alır.
- *Alfa Testi:* Yazılım geliştirmelerin yani kodlamanın sonlanmasına yakın geliştirmenin yapıldığı ortamda gerçekleştirilen testlerdir.
- *Beta Testi:* Yeterli bir olgunluğa gelen yazılımın son kullanıcıları ile kullanıcının kendi (test ya da canlı) ortamında yaptığı testlerdir.

3.4.Yazılım Test Tasarım Teknikleri

Test tekniklerinin amacı test koşullarını, test verilerini ve senaryolarını tanımlamaktır. Test teknikleri kara kutu ve beyaz kutu test teknikleri olarak ikiye ayrılabilir. Kara kutu test tekniklerine spesifikasyon bazlı test teknikleri adı da verilir. Test şartlarını, durumlarını ve verilerini üretmek için test dokümanlarının analizine dayanan bu testler fonksiyonel ve fonksiyonel olmayan testleri içerir. Beyaz kutu test tekniği diğer adıyla yapısal veya yapı bazlı test tekniği, bileşen veya sistemin içyapısına odaklanarak kodların test edildiği test

tekniklerdir. Test edilmesi gereken ya da önceliklendirilmesi gereken durumlara karar verilmesi için yazılımcıların, test uzmanlarının ve kullanıcıların tecrübelerini kullanmak amacıyla kara kutu ve beyaz kutu testi, deneyim temelli tekniklerle de bir araya getirilir. Bunlar saydam kutu, kara kutu ve gri kutu test teknikleridir [21].

3.4.1. Kara kutu test teknikleri

Yazılımın fonksiyonlarının içyapısı (kod, database, aktarım vb.) görülmediğinden kullanıcı gibi gerçekleştirilen kutu testleri yazılımın tüm seviyelerinde kullanılabilir ama fonksiyonların ortaya çıkması gereklidir. Kullanıcı kabul testleri ve sistem testlerinde gereksinimler referans alınarak, entegrasyon ya da birim testlerinde analiz dokümanları ya da tasarım dokümanları baz alınarak gerçekleştirilir. Aşağıda kara kutu test teknikleri bulunmaktadır:

- *Sınır Değer Analizi (Boundary Value Analysis):* Sınır değerlere bakılarak (örneğin, gereksinimler içerisinde 38 yaş sınırı var ise 37, 38 ve 39 girdilerine karşı üretilecek çıktıların testi) test edilir. Değişimin olacağı durumlarda yazılımın karalılığını test etmek üzere yapılır. En büyük ve en küçük değişim noktaları tespit edilir ve eşdeğer aralıklar çıkartılır.

- *Karar Tablosu Testi (Decision Table Testing):* Test edilecek koşulların fazla olması halinde mevcut durum bir tablo üzerine aktarılarak karar tablosu oluşturulur. Her bir duruma karşı yazılım vereceği cevaplar tabloya yerleştirilir.

- *Eşdeğer Aralık Testi (Equivalence Partitioning*): Bir kara kutu test tekniği olan bu teknik aynı değeri üreten aynı bölgedeki değerler girdi olarak kullanılırsa aynı sonuçlar yani aynı çıktılar üretir koşuluna göre çalışır. Gereksinimler içerisinde 10 yaş sınırı varsa 10 yaşına kadar olanlar için hayır, 10 dan sonrası için evet üretecekse 10 ve 20 kullanılabilir.

- *Durum Geçiş Testi(State Transition Testing*): Geçiş durumlarında ortaya çıkabilecek sorunlar test edilir.

- *Fayda Analizi Testi (Use Case Testing):* Kullanım durumu bir oyuncu veya kullanıcı tarafından kullanıcının tüm hareketine karşılık sistemin nasıl davrandığı geçerli durum olarak kaydedilir. Sistem veya kabul testlerinde kullanılır[21].

3.4.2. Beyaz kutu test teknikleri

Yazılımın iç dinamikleri (mimari yapı, SQL sorguları, mantık yapıları, algoritmalar vb.)dikkate alınarak analiz çıktılarının yazılımcılar tarafından test edildiği bu test tekniği kullanılarak gerçekleştirilen test türleri aşağıdadır:

Kod kapsamı (code coverage)

Toplam çalıştırılan test durumlarının çalıştırılması gereken toplam test durumlarının oranı olan kapsama, yazılımda ne kadar test yapıldığını anlamak üzere geliştirilmiş metrikler anlamına gelir [21].

Kapsam=(Çalıştırılan Test Durum Sayısı/Toplam Test Durum Sayısı)X100

Kontrol noktalarının kapsamı (Decision Coverage)

Toplam çalıştırılan döngülerin yazılan koddaki tüm döngülere oranı ile bulunan bu kapsam, kod içerisinde ne kadar döngünün çalıştırıldığını göstermek için kullanılan birimdir.

Tüm noktaların kapsamı (Statement Coverage)

Bu birimde geliştirilen koddaki ifadelerin çalıştırılma oranı bulunur.

3.4.3. Deneyim temelli test teknikleri

Hata tahmini ve keşif temelli testlerin yapıldığı test tekniğidir. Deneme yanılma ile yazılımın verdiği cevapların mantığa uygunluğu kontrol edilerek gerçekleştirilen bu yöntemde testçinin tecrübeli olması ön plandadır ve daha önce benzer uygulamaları test etmiş olması önemli bir faydadır. Keşif temelli testlerde ise analizin zayıf veya olmadığı ve zamanın kısıtlı olduğu durumlarda yapılabilecek bir test tekniklerini tamamlayıcı olarak uygulanan bir davranıştır. Test sırasında dokümantasyon bulunmamaktadır sadece bulunan hatalar

raporlanır ve testler manuel işletilir. Yazılımın keşfinin yapıldığı keşif temelli testler yazılımın güçlü, zayıf olan taraflarını bulmak, yazılımın neler yapıp yapmadığını ve nelerin yanlış olduğunu öğrenmek üzere yapılır [21].

3.5. Yazılım Test Türleri

Yazılımların yapısı arttıkça yazılımlarda ortaya çıkabilecek olan hata sayısı artar. Yazılımlarda karşılaşılan hata, yazılımların bitirilme zamanlarında gecikmelere, maliyetin artmasına ve geliştirilen yazılımın başarısız olmasına neden olabilir. Bunun için hataların, kusurların ve eksikliklerin yazılım geliştirme yaşam döngüsünün ilk aşamalarında tespit edilmesi, bunların düzeltilmesi için faaliyetlerin yapılması ve bu faaliyetlere göre yazılım geliştiriminin devam etmesi gereklidir. Yazılım yaşam döngüsü boyunca gerçekleştirilen test faaliyetleri statik testler ve dinamik testler olmak üzere iki başlık altında toplanır [17].

3.5.1. Statik testler

Statik testlerde kod yürütülmeden önce kodun ve proje kapsamında bulunan dokümanların gözden geçirilmesine dayanan testlerdir. Dinamik testler yapılmadan yazılımın kod dâhil test edildiği bu testler, yazılım geliştirme süresince kod seviyesinde olan ve geliştiricilere destek vererek hataların önceden tespit etmelerine yardımcı olabilen tekniklerdir. Herhangi bir kod çalıştırılması olmadan yazılı olan her şeyin gözden geçirilmesi süreci dâhil edilerek çıktılar üzerindeki hatalar önceden tespit edilir. Statik test teknikleri ile oluşabilecek hatalar önceden tespit edilebilir ama dinamik test tekniklerinin yerine geçemez [21].

3.5.2. Yazılım gözden geçirmeleri

Statik test teknikleri içerisinde yer alan gözden geçirme süreçleri kod, doküman, gereksinimler uzman kişilerce gerçekleştirilerek daha sonra ortaya çıkabilecek hataların ayıklanması sağlanır [21].

Gözden geçirmenin faydaları;

- Daha az defect ve yazılım maliyeti

- Yazılım geliştirme faaliyetlerinde artan verimlilik
- Yazılım geliştirmelerde kısalan süreler
- Hata seviyesinin düşürülmesi, daha az ciddi hataların kalması
- Müşteri ilişkilerinin artması
- Deneyim paylaşımının artması
- Standartlara uyumun artması

Gözden geçirme ve test süreçleri

- *Bireysel Gözden Geçirme Yöntemleri:* Bireysel olarak yapılan bu yöntem yazılımcının kendisinin yaptığı ve grupla yapılacak gözden geçirmelerden önce kendi hatalarını azalttığı olgunlaşmış bir kod ortaya çıkardığı yöntemdir.
    - *Masada Kontrol(Desk Checking):* Yazılan kodun çıktısı alınarak masa başında satır satır kod üzerinde geçerek simüle edilir. Mantık sorguları kontrol edilir ve hatalar bulunur.
    - *Uzman Deneyimi(Proof Reading):* Uzmanlar tarafından hazırlanmış kod, algoritma vb. dokümanlar okunarak kendi çıktıları test edilir.
- *Grup Gözden Geçirme Yöntemleri:* Uzmanlaşmış kişilerce yapılan bu yöntemde sistematik bir şekilde ön yargılardan uzaklaşarak başka kişilerin yakalayabileceği hataların olacağı durumlar ortadan kaldırılır. Aşağıdaki yöntemlerden oluşur:
    - *Gözden Geçirme (Reviews):* Resmi veya gayrı resmi yapılan bu yöntem genelde grup olarak yapılır ve bu yöntemde resmi yapılan gözden geçirmeler hazırlıklı olur ve zamanlaması önceden planlanır. Konuyu değerlendiren uzmanların kendi fikirlerini ortaya koyduğu ve karar çıkması durumunda uygulamaya geçildiği yöntemdir. Gayrı resmi gözden geçirmede ise gözden geçirmeler hazırlıksız ve kısa görüşmeler ile gerçekleşir.
    - *Üzerinden Geçme(Walkthrough):* Planlı ve öncesinde katılımcıların hazırlanarak katıldığı bu görüşmelerde dokümanın veya kodun sahibi toplantılarda çıktıların üzerinden geçer.
    - *İnceleme(Inspection):* Planlı, dokümanlar ve araçlar yardımı ile yapılan bu yöntemde saptanan hatalar raporlanır ve düzeltilmesi için yazılımlara iletilir. Böylece inceleyiciler ile kodu yazan ortak karara vardıklarında süreç

tamamlanır ve onaylanan kod, doküman bir sonraki sürece geçer. Resmi (formal) gözden geçirmelerde süreç aşağıdaki şekildeki gibi ilerlemektedir.

Gözden geçirme sürecinde roller

Gözden geçirme sürecinde genel olarak aşağıdaki rollerin bulunması gerekmektedir:

- *Lider/Moderatör:* Gözden geçirme toplantılarını yöneten kişilerdir.
- *Yazar:* Yazılım geliştiricisi, doküman sahibi
- *İnceleyici:* Yazılım kodu, proje dokümanı vb. inceleyen uzmanlaşmış kişiler
- *Yönetici:* Planlamayı yapan, yöneten ve iletişim kuran kişi
- *Yazıcı:* Toplantı sırasındaki tüm bulguları, gereklilikleri kaydeden kişi.

3.5.3. Araçlarla gerçekleştirilen statik kod analizleri

Araçlarla yapılan statik kod analizleri ile hatalar önce bulunarak, karmaşık kodlar bulunarak, dinamik testlerde bulunması zor olan hatalar ve tutarsızlıklar belirlenir. Analizde kullanılan araçlar programları çalıştırmadan sadece analiz yaparlar. Analiz edilecek kod parçaları, html veya xml gibi dosyalarıdır. Statik analizde saptanan hatalar yazılımın içyapısıyla ilgilidir ve bulgunun önemi düzeltilmediği durumda yazılıma maliyeti çok fazladır. Kullanılan yazılım diline göre değişebilen bu tür araçları kullanmak beraberinde yazılım dilini iyi bilmeyi zorunlu hale getirir ve bu nedenle araçlar geliştiriciler tarafından kullanılır. Bu araçlar;

- *Derleyici(Complier):* Kodların derlenmesi ve çalıştırılması sağlanır.
- *Cyclomatic Complexity:* Kodun karmaşıklık seviyesini gösteren araçlardır.
- *Akış Şeması(Control Flow):* Kodun yürütülmesi esnasında neleri yapacağını gösteren yol akış şeması.
- *Veri Akışı(Data Flow):* Verilerin oluşturulması, kullanılması ve silinmesi adımlarını soyut şekilde anlatan akış şemalarıdır.
- *Statik Analiz(Static Analysis):* Yazılımı oluşturan parçaların çalıştırılmadan analiz edilmesidir.

Statik analiz araçları tarafından bulunan hatalar aşağıdaki gibi sıralanabilir:

- Değer atanmamış değişkenin istenmeden referans gösterildiği durumlar
- Modüller ve bileşenler arasında tutarsız ara yüzler
- Kullanılmayan ve yanlış tanımlanmış değişkenler
- Karmaşık yapılar
- Güvenlik açıkları
- Eksik ve hatalı mantık
- Ulaşılmayan, çağrılmayan kod parçaları
- Kodlarda ki söz dizimi hataları

Statik analizin avantajları:

- Testler yürütülmeden önce hataların erken tespiti yapılır.
- Karmaşıklık ölçüsü fazla olan koddaki veya tasarımdaki sonuçlarla ilgili erken uyarı sağlanır.
- Kodun iyileştirilmesi ve tasarımın sürdürülebilir olması sağlanır.
- Uyarılar geliştirme sırasında dikkate alındığında hataların önlenir.

3.5.4. Dinamik testler

Kodlamanın sonlanmaya başladığı yani yazılım ürünü olgunlaşmaya başladığında testlere başlanan dinamik testler, geliştirilen yazılım ürünün dinamik olarak nasıl davrandığının gözlemlenmesi için gerçekleştirilen testlerdir. Dinamik testlerde sisteme girilen değişik veriler sonrasında nasıl tepki verdiği gözlemlenir. Dinamik testler, testlerin koşturularak yapıldığı hataların bulunduğu test aktiviteleridir.

4.TEST SÜREÇ İYİLEŞTİRME ve TEST OLGUNLUK MODELLERİ

Bir yazılımın kalitesi, açık ve net bir şekilde tanımlanmış gereksinimlere uyum, müşteri isteklerini karşılayabilmesi olarak tanımlanabilir. Ayrıca açık bir şekilde dokümantasyonu yapılmış yazılım gelişme standartlarına bağlı kalma, kullanıcı ve diğer paydaşlar açısından güvenilirlik sağlama, yazılımda farklı ve çeşitli özelliklere sahip olma ve ürün teslim sonrası ve tatmin edici destek sağlama olarak tanımlanabilir [22].

Test süreci, hata tespiti ile ilgili aktiviteleri gerçekleştirmenin belirli bir yoludur. Bu tür birkaç faaliyet aşağıdaki gibidir:

- Test hedefleri belirleme
- Test planı hazırlama
- Farklı test seviyeleri tanımlama
- Test personeli atama
- Test senaryoları tasarlama
- Test ortamlarını kurulması
- Test araçları temini ve kullanılması
- Test durumlarının yürütülmesi vb.

Sistem düzeyinde basit bir test süreci için sistemin test edilecek özelliklerine sınıflandırma yapılır, bu kategoriler için test durumları oluşturulur, oluşturulan test durumları hataları tespit etmek için koşturulur, hataların düzeltilmesi ile test sonlandırılır. Bu basit süreç için eksiklilere bakılırsa test araçları kullanılmaması, test durumlarının önceliklendirilmemesi göze çarpan eksikliklerdir. Bu nedenle, tanımlanmış bir modeli takip ederek test işlemlerini geliştirmek önemlidir. Test süreci iyileştirmesi için bir modeli takip ederek test süreçlerini geliştirme fikri ilk kez Tim Koomen ve Martin Pol tarafından incelenmiştir. Bir test sürecinin aşağıdaki üç sebepten dolayı iyileştirilmesi gerekir:

- Kalite: Daha iyi bir test süreci, test edilen sistemin kalite özellikleri hakkında daha fazla bilgi vermelidir.

- Teslim Süresi: Daha iyi bir test süreci, test teslim takviminde gecikmelerin önüne geçer.
- Maliyet: İyi bir test sürecinin daha düşük bir maliyetle gerçekleştirilmesi beklenir.

Bir test süreci geliştirmek için sezgisel bir yaklaşım aşağıdaki gibidir:

- Adım 1: İyileştirilecek alan belirlenir.
- Adım 2: Test süreci mevcut durumu belirlenir.
- Adım 3: Bir sonraki istenen durumu ve araçları tanımlanır.
- Adım 4: İşlem için gerekli değişiklikler uygulanır.

4.1. Test Süreç İyileştirmenin Yararları

Test olgunluk modelleri kullanarak test süreçlerinde iyileştirme çalışmalarında bulunacak şirketler için getirecek faydaları aşağıdaki gibi sıralanabilir:

- Test faaliyetleri etkinliği ve verimliliği arttırılır.
- Testleri organizasyon öncelikleri ve diğer proje süreçleriyle uyumlu hale getirilir.
- Kuruluş için testin değeri arttırılır.
- Genel yazılım kalitesi arttırılır.
- Zaman kaybı engellenir.
- Hatalar azaltılır.
- Uyumluluk geliştirilir.
- Daha verimli ve etkili iş operasyonları oluşturulur.
- Test maliyetlerinde uzun vadeli azalmalar gerçekleşir.

4.2. Test Süreç İyileştirme ve Olgunluk Seviyesi Modelleri

Test işlemleri, bir yazılım ürünündeki olası hataların bulunup, düzeltilmesini amaçlamaktadır. Yazılım sistemlerinde ise, yaşam döngüsü modeline göre bu işlemler farklı

şekillerde yapılmaktadır. Test süreçlerinin olgunluk seviye ölçümü için farklı metodolojiler kullanılabilmektedir[5]. Çizelge 4.1' de verilmiş olan bazı modeller aşağıda verilmiştir.

Çizelge 4.1. Test süreç iyileştirme modelleri [5]

Kısaltma	**Detay**	**Yazar/Organizasyon**
BTM	Beizer' in Test Modeli	1990 yılında Beizer tarafından hazırlanmıştır.
TOM	Test Organizasyonu Olgunluk Modeli	1990'ların sonunda İngiltere'deki Gerrad Consulting tarafından yazılmıştır.
TMAP	Test Yönetimi Yaklaşımı	Martin Paul, Rudd Teunissen ve Erik tarafından yazılmıştır 1995 yılında Veenendaal olabilir ve şu anda Capgemini'nin bir parçası olan Sogeti'ye aittir.
TSM	Test Edilebilirlik Destek Modeli	1996 yılında Dr. David Gelperin tarafından yazılmıştır.
TPI	Test Süreci İyileştirme Modeli	1996 yılında Tim Koomen ve Martin Pol tarafından yazılmıştır.
TMM	Test Olgunluğu Modeli	1996 yılında Illinois Institute of Technology'de Dr. Ilene Bernstein tarafından yazılmıştır.
TCMM	Teknik Yetenek Olgunluk Modeli	2001 yılında Torry Harris tarafından yazılmıştır.
TIM	Test Geliştirme Modeli	2002 yılında Thomas Ericson tarafından yazılmıştır.
TCI	Test Kabiliyetini Geliştirme	Atos Origin tarafından hazırlandı
TPA	Test Süreci Değerlendirmesi	CTG tarafından hazırlandı
TMAP Next	Yeni Nesil İçin Test Yönetimi Yaklaşımı	2005 yılında Tim Koomen, Michiel Vroon, Leo van der Aalst ve Bart Broekman tarafından yazılmıştır.
TMMi	Entegre Test Olgunluğu Modeli	2007 yılında TMMi Vakfı tarafından. 2010 yılında 4. ve 5. seviye çerçeveler yayınlanmıştır.
TPCR	Test Süreci ve Yetenek Değerlendirme	2011 yılında Barış Sarıgerioğlu tarafından hazırlanmıştır ve 2013 yılında revize edilmiştir.

Genel olarak test süreç iyileştirmesi için tanımlanan modeller aşağıdaki tabloda verilmiştir. Bunların içinden dünya çapında tanımlanan modeller TPI ve TMMi modelleridir. TMMi modeli CMMI modelinin tamamlayıcı nitelikte olması ve her olgunluk seviyesi için aşamalı ve uyması gereken kriterleri açıkça tanımlamaktadır. Sonraki seviyeye geçilmesi için düşük seviyedeki süreç alanlarının karşılanması beklenir [5]. Aşağıda verilmiş olan modellerden bazılarına yer verilmiş TMMi modeli ilerleyen bölümlerde detaylı olarak ele alınmıştır.

4.2.1. Test süreci ve yetenek değerlendirme modeli (TPCR)

Barış Sarıalioğlu tarafından 2011 yılı itibariyle geliştirmeye başlanan TPCR'ın yapısı ve prensipleri ile diğer modellerden ayıran en büyük özelliği yalnızca test süreçlerine odaklanmamasıdır. Bunun yanında ilişkili diğer süreçleri de inceliyor olmasıdır. Bu süreçler

Test Processes altında gruplanmıştır. Bu kapsamda 32 kritik alan ve bu alanların dört başlık altında listesi aşağıdaki tablodaki gibidir[5].

Çizelge 4.2. Test süreci ve yetenek değerlendirme modeli kritik alanları [5]

Kısaltma	Anahtar Süreç Alanı
TP	**Test Süreçleri**
TP.01	Test Metodolojisi
TP.02	Test Politikası
TP.03	Test Planlama ve Bütçeleme
TP.04	Test Efor Tahmini
TP.05	Erken Katılım
TP.06	Test Temelleri
TP.07	Test Durdurma Kriterleri
TP.08	Risk Değerlendirmesi
TP.09	Test Metrikleri
TP.10	Test Dokümantasyonu
TP.11	Test İzleme ve Raporlama
TP.12	Test Tasarım Teknikleri
TP.13	Statik Test
TP.14	Fonksiyonel Test
TP.15	Fonksiyonel Olmayan Test
TP.16	Regresyon Test
TP.17	Geliştirme ve Birim Testi için Test Faaliyetleri
TP.18	Kullanıcı Kabul Testi
TP.19	Olumsuz Test
PO	**İnsanlar ve Organizasyon**
PO.1	Kişiler / Organizasyon
PO.2	Test İçin Proje Rolleri
PO.3	Test Becerileri
PO.4	Test Eğitim Programı
TT	**Teknoloji Araçları**
TT.1	Test Araçları ve Kullanımı
TT.2	Test Ortamları
TT.3	Test Otomasyonu
TT.4	Test Veri Yönetimi
RP	**İlgili İşlemler**
RP.1	Hata Yönetimi
RP.2	Yapılandırma Yönetimi
RP.3	Yayın Yönetimi
RP.4	Gereksinim Yönetimi
RP.5	Kullanıcı Deneyimi ve Kullanılabilirlik

TPCR modeli tabloda verilmiş alanlardaki olgunluğun ölçümlenmesini ve iyileştirilmesine katkı sağlanmasını amaçlamıştır. Modelde 10'luk skala üzerinden derecelendirme bulunmaktadır. Bu derecelendirme sonucunda 0 ile 10 arasında aşağıdaki gibi puanlamalar yapılmaktadır [5].

Geçici[0,2): Ad Hoc seviyesinde test süreçleri manuel gerçekleştirilmektedir. Bu nedenle gerçekleştirilen aktiviteler tekrar edilememektedir.

Benimsenmiş[2-4): Test aktivitelerinin bir süreç olarak kurgulandığı bu seviyede tanımlı test planı, test strateji dokümanı ve gereksinimlerden oluşan test senaryoları bulunmaktadır. Bu seviyede test işlemlerinin başlayabilmesi için ürünün test aşamasına gelmesi beklenir.

Ölçülü[4-6): Yazılım geliştirme yaşam döngüsüne test işlemleri entegre edilmiştir. Risk yönetimine dikkat edilir ve test işlemleri bu seviyede geliştirme aktivitelerinden bağımsızdır.

Gelişmiş[6-8): Bu seviyede test aktiviteleri, müşteri isteklerinin ve tasarımın geçerlilik kazanması dahil tüm yaşam döngüsü aşamalarında yer almaktadır. Kalite kriterleri tüm organizasyon seviyesinde kabul edilmiştir.

Standartlaştırılmış[8-10): Test aktivitelerinin kontrolü süreklilik kazanmakta ve geliştirilmektedir. Bu seviyede genel olarak bir araç desteği ve hata önleme işlemleri bulunmaktadır.

4.2.2. Test olgunluğu modeli (TMM)

Burnstein ve diğerlerinin Illinois Teknoloji Enstitüsünde 1996 yılında Test Olgunluk Modelini (TMM) geliştirmelerinin ana nedenlerinden biri, mevcut olgunluk modellerinin test konularına yeterince cevap vermemesi ve olgun bir test işleminin doğası iyi tanımlanmamasıdır. Olgunluk seviyeleri ve alt hedefleri Burnstein ve arkadaşları tarafından geliştirilmiştir. Suwannasart 1996 da TMM için temelleri atmıştır. Literatürü kapsamlı araştıran Suwannasart TMM' ye faaliyetler, görevler ve sorumluluklar(uygulamalar) tanımlamıştır. TMM'in amacı, bir kuruluş içindeki değerlendirme ve iyileştirme kullanıcılarını desteklemektir. TMM modeli aşağıdakiler tarafından kullanılabilir:

- Mevcut yetenek durumunu belirlemek için bir iç değerlendirme ekibi.
- Bir test iyileştirme programı başlatmak için üst yönetim.
- Süreç iyileştirme planları geliştirmek ve uygulamak için yazılım kalite güvence mühendisleri.
- Test etkinliği geliştirmek için geliştirme ekipleri.
- Test sürecinde rollerini tanımlamak için kullanıcılar ve müşteriler.

TMM modeli olgunluk modeli ile değerlendirme prosedürü, değerlendirme anketi, ekip eğitimi ve seçim kriterleri içeren değerlendirme modelinden oluşur.

4.2.3. Yeni nesil için test yönetimi yaklaşımı modeli (TMAP Next)

TMap NEXT 2006 yılında yayınlanan ve hala süreç odaklı birçok kurum için standart test yöntemi olarak kullanılan yapılandırılmış bir yaklaşımdır. Aşağıdaki avantajlara sahiptir:

- Test edilen sistemin kalitesiyle ilgili herhangi bir risk hakkında fikir verir ve tavsiyelerde bulunur.
- Erken aşamada hataları bulur.
- Hataları önler
- Test ürünleri tekrar kullanılabilir.
- Test süreci anlaşılabilir ve yönetilebilir.

Yapılandırılmış bir test yaklaşımının spesifik TMap içeriği dört temelde aşağıdaki gibi özetlenebilir.

- TMap, İş Odaklı bir Test Yönetimi (BDTM) yaklaşımına dayanmaktadır.
- TMap, yapılandırılmış bir test sürecini açıklar.
- TMap eksiksiz bir araç kutusu içerir.
- TMap uyarlanabilir bir test yöntemidir.

4.2.4. Test geliştirme modeli (TIM)

Test iyileştirme modeli, Ericson, Subotic ve Ursing tarafından geliştirilmiş. CMM ve Gelperins TSM'ye dayanmaktadır. TIM modeli, test çalışmalarını iyileştirmeye yönlendirmeyi amaçlar. Geliştiricilere göre TIM iki şekilde kullanılabilir:

- TIM, kilit alanlarda mevcut uygulama durumunu belirlemek için kullanılır.
- TIM, güçlü yönler üzerine inşa edilebilecek zayıf yönleri ortadan kaldırılabilir.

TIM şunlardan oluşur:

- Olgunluk modeli
- Değerlendirme prosedürü

4.2.5. Test organizasyonu olgunluk modeli (TOM)

System Evolutif'te geliştirilen model yetenek olgunluk modelinin geliştiricilere göre sorunları iyi karşılamaması, çare temelli kullanılan yaklaşımların yetersiz olması üzerine geliştirilmiştir. Geliştiricilere göre gelişmiş uygulamaların önündeki başlıca engeller teknik değil organizasyoneldir. Geliştirilmiş uygulamaların uygulanmasındaki zorlukların çoğu, çalışanların süreçleri ve yönetim kontrollerini değiştirmek ve uygulamak için doğal direncinin üstesinden gelmek, değişen yönetim algılarıyla ilişkilidir. TOM modelinin amacı, kurumsal darboğazları desteklemek, tanımlamak ve önceliklendirmek ve bu sorunlara çözüm üretmektir. Model semptomları belirleme ve önceliklendirme anketi ile iyileştirme önerilerinden oluşur.

4.2.6. Test süreçleri iyileştirme modeli (TPI)

TPI modeli 1997 yılında Koomen ve Pol tarafından geliştirilmiştir. Modelin geliştirilmesinin ana nedenlerinden biri, testin önemli ancak zor ve kontrolsüz süreç olarak görülmesidir. TPI modeli, test sürecinin iyileştirilmesini kolaylaştırmak için geliştirilmiştir. Koomen ve Pol, 1999 yılında TPI hakkında bir kitap yayınlamışlardır. TPI modeli, bir kuruluştaki test sürecinin zayıf ve güçlü alanlarını belirlemek için bir çerçeve sunar [6].

Ek olarak, sürecin olgunluk seviyesi belirlenir ve TPI modeli iyileştirme faaliyetlerinin benimsenmesini destekler. TPI modeli aşağıdaki parçaları içerir:

- Olgunluk modeli
- Test olgunluk matrisi
- Kontrol listesi
- İyileştirme önerileri

5.BÜTÜNLEŞİK YETKİNLİK OLGUNLUK MODELİ (CMMI)

Tez çalışmamızda kullandığımız TMMi modeli aslında CMMI modelinden esinlenerek oluşturulmuştur. Giriş bölümünde belirtilen Bütünleşik Yetkinlik Olgunluk Modeli (CMMI) yazılım sektöründeki şirketlerin planlama, geliştirme, konfigürasyon, bakım ve onarım gibi süreçlerinin olgunluk derecesinin ölçüldüğü değerlendirme modelidir. Yazılım geliştirme sektöründe bulunan şirketlerin belirli periyotlar da yeteneklerinin geliştirildiği ve seviyelerinin belirlendiği bir model olarak kullanılır. 1986 yılında Yazılım Mühendisliği Enstitüsü tarafından geliştirilen Yetenek Olgunluk Modelinin (CMM), 1991 yılında ilk sürümü çıkmıştır. Enstitünün bu sistemi şu anda 2000'li yıllarda ortaya çıkan ve CMMI olarak bilinen ve yazılım mühendisliği yetkinlik hedeflerinin sağlamlaştırılmasında yazılım şirketleri tarafından kullanılmaktadır. Bu model ile yazılım projesi gereksinimlerine ve uygunluğuna göre, tüm organizasyonun ya da belirli birimlerin iyileştirilmesi ve geliştirilmesi temel alınır. Yazılım geliştirme şirketleri CMMI modelinde bulunan beş aşamalı seviyelerden 1 ile 5 arasında değerlendirilirler. CMMI modeli bağımsız ilerleyen organizasyonel birimlerin uyumlu hale getirilmesine, süreç iyileştirme ve önceliklerin belirlenmesine, sürecin kaliteli hale getirilmesine katkıda bulunur.
Aşamalı değerlendirme ise belirli adımlar içerir.

5.1. Başlangıç: Seviye 1

Yazılım geliştirmenin bir sürece bağlı kalmadan yapıldığı bu seviyede sürecin belirlenen amaçlarından çoğunun karşılanmadığı bir durum söz konusudur. Bu seviyede başarı kişisel çabalar ve çalışanların tecrübeleri ile oluşmaktadır [1].

5.2. Yönetilen: Seviye 2

Proje yönetimi ile ilgili kontrol noktaları belirlenmiştir ve projeler önceden belgelenmiştir. Proje planlarına göre yapılması gereken işler yapılmaktadır fakat kurumsallaşma henüz karşılanamamıştır. Proje yönetiminde bulunan sistem ile proje geliştirme süreci kontrol edilir. Projeler bu seviyede planlanabilmekte, başarılı olabilmekte, ölçülebilmekte ve kontrol edilebilmektedir. Yapılan anlaşmalar, ilgilenilen projedeki bütün paydaşların talep ve değişiklik istekleri ile uyumludur [1].

5.3. Tanımlanmış: Seviye 3

Yazılım geliştirme süreçlerinin tanımlandığı ve tüm paydaşlar tarafından anlaşıldığı bu seviyede standartlar, prosedürler, araçlar ve metodolojileri bulunur. Standartlaştırılmış süreç tanımları organizasyon içinde tutarlı ve uyumludur. Bu seviyede bütün projelerde tanımlanmış standart yazılım geliştirme süreçleri kullanılır. Süreçler ikinci seviyeye göre daha özenli ve detaylıdır [1].

5.4. Nicel Olarak Yönetilen: Seviye 4

Nicel olarak yönetilen seviyede istenen kalite açısından hedeflenen ölçümlere ve genel olarak sürecin performansına geçilmiştir. Sürecin başarısızlığı ya da başarısı ölçüm teknikleri kullanarak belirlenir. Kalite ve süreç performansı için nicel hedefler belirlenir ve bunlar süreçlerin yönetiminde bir ölçüt olarak kullanılır. Nicel hedefler, müşterilerin ihtiyaç ve istekleri, ürünü kullanacak olan son kullanıcılar, organizasyondaki kişiler ve bu süreçleri geliştirenler temel alınarak belirlenir. Üçüncü seviyeden farklı olarak bu seviyede süreç performansı öngörülebilir durumdadır. Bu seviyede, elde edilen veriler istatistiksel olarak ve nicel teknikler ile kontrol edilebilir ve sonraki aşamalar için öngörülebilir [1].

5.5. Optimizasyon: Seviye 5

Nicel olarak yönetilen seviyede ulaşılan veriler ve projelerin tamamlandığı süreçte teknoloji kullanılarak bu seviyede sonuçlanan bütün projelerin iyileştirilmesine ve bu projelerden en yüksek seviyede yarar sağlanmaya odaklanılır.

Bu seviyede, süreçten elde edilen ve teknolojilerden yararlanılarak sürekli iyileştirilme yapılır. Organizasyondaki tüm çalışanların süreç iyileştirmeye odaklandığı beşinci seviyede yazılım geliştirme süreci sürekli iyileşen olarak tanımlanabilir. Dördüncü seviyeden farklı olarak sürekli iyileşen ve gelişen bir durumda yeniliklerin olduğu süreç performansı gözle görülür şekilde etkisini gösterir [1].

6.TEST OLGUNLUK MODELİ ENTEGRASYONU (TMMi)

Bu bölümde TMMi vakfı tarafından yayınlanmış olan belge özetlenecektir [19].

6.1. TMMi Tarihçe

Çalışmanın esasını oluşturan Test Olgunluk Modeli Entegrasyonu (İngilizce "**T**est **M**aturity **M**odel **i**ntegration" ifadesinin kısaltması olan TMMi), Yetenek Olgunluk Modeli Entegrasyon (İngilizce "**C**apability **M**aturity **M**odel **I**ntegration" ifadesinin kısaltması olan CMMI) modeline dayanarak TMMi vakfı tarafından geliştirilen bir modeldir. Test işlemlerinin olgunluğunu belirlemek amacıyla bir sistem oluşturan ve testin olgunluğunu geliştirmek için hedefler belirleyen bu modelde tanımlanan testler, tüm yazılım ürün kalite ile ilgili faaliyetleri kapsayacak şekilde en geniş anlamıyla uygulanmaktadır. Süreç değerlendirme ve iyileştirme için CMMI modelinde olduğu gibi olgunluk seviyesini kullanır. Ayrıca süreç alanları, süreç hedefleri ve süreç uygulamaları belirlenir. TMMi olgunluk seviyelerini uygulamak, test sürecini iyileştirecek ve ürün kalitesi, test mühendisliği verimliliğini ve zaman harcamasını olumlu yönde etkileyecektir. TMMi, test sürecini değerlendiren ve geliştiren kuruluşları desteklemek için geliştirilmiştir. TMMi ile birlikte testler, bir meslek haline gelir ve yazılım geliştirme yaşam döngüsüne tamamen entegre olan bir durumdadır. Testin odak noktası hata tespitinden ziyade hata önleme değişlikleri haline gelir.

Gelperin ve Hetzel'in evrimsel test modeli TMMi 'de tarihsel düzeydeki farklılaşma için bir temel oluşturmuştur. TMMi, Illinois Institute of Technology tarafından geliştirilen TMM3 çerçevesini ve BT sektöründeki yaygın desteğe sahip bir süreç iyileştirme modeli olan Yetenek Olgunluk Model Entegrasyonu (CMMI) kaynak alarak oluşturulmuştur. TMMi modelinde kullanılan test terminolojisi, Uluslararası Yazılım Test Yeterlilik Belgesi (ISTQB) standart terimler sözlüğünden türetilmiştir.

TMMi modeli, sistem mühendisliği ve yazılım mühendisliği disiplinlerinde test aktivitelerini ve test sürecini iyileştirmeyi desteklemeyi amaçlamaktadır. Sistem mühendisliği, yazılımı içerebilen veya içermeyen toplam sistemlerin geliştirilmesini kapsar. Yazılım mühendisliği yazılım sistemlerinin geliştirilmesini kapsar.

6.2. TMMi Seviyeleri

TMMi modelinde bir olgunluk hiyerarşisi ve süreci iyileştirmeyi test etmek için evrimsel bir yol belirleyen beş seviye vardır. Her seviye, bir kuruluşun bu düzeyde olgunluğa ulaşmak için uygulamaya koyması gereken bir dizi süreç alanına sahiptir. Her olgunluk seviyesi bir sonraki seviye için gerekli bir temel oluşturur.

Çizelge 6.1. TMMi olgunluk seviyeleri ve süreç alanları

BAŞLANGIÇ	L2-YÖNETİLEN	L3-TANIMLI	L4-ÖLÇÜLEN	L5-OPTİMİZE
Süreç alanı bulunmamaktadır.	Test Politikası ve Stratejisi	Test Organizasyonu	Test ölçümü	Hata Önleme
	Test Planlama	Test Eğitim Programı	Yazılım Kalite değerlendirmesi	Test Süreç Optimizasyonu
	Test İzleme ve Kontrol	Test Yaşam Döngüsü ve Entegrasyonu	Gelişmiş Eş Gözden Geçirmeler	Kalite Kontrol
	Test Tasarım ve Yürütme	Fonksiyonel olmayan testler		
	Test Ortamı	Eş Gözden Geçirmeler		

TMMi modelinin her bir olgunluk seviyesi için süreç alanları Çizelge 6.1'de gösterilmektedir. Bunlar daha sonra diğer bölümlerde açıklanmıştır ve her TMMi seviyesinde bir organizasyonun özelliklerinin kısa bir açıklaması ile birlikte aşağıda listelenmiştir.

6.2.1. Başlangıç: Seviye 1

TMMi seviye 1'de test tanımlanmamış bir süreçtir ve genellikle hata ayıklamanın bir parçası olarak kabul edilir. Bu organizasyonlarda başarı, kanıtlanmış süreçlerin kullanılmasına değil, organizasyondaki insanların yetkinliğine bağlıdır. Kodlama tamamlandıktan sonra testlere başlanır. Test ve hata ayıklama, hataları sistemden çıkarmak için birlikte gerçekleştirilir. Bu seviyede test etme amacı, yazılımın büyük hatalar olmadan çalıştığını göstermektir. Ürünler kalite ve riskler konusunda yeterli görünürlük olmadan canlıya alınır. Test kapsamında kaynak, araç ve iyi eğitimli personel eksikliği var. TMMi Seviye 1'de tanımlanmış bir süreç alanı yoktur. Olgunluk seviyesi 1 olan şirketler, krizlerin gerçekleştiği süreçlerde aşırı işlem yapma, işlemlerden vazgeçme ve başarılarını tekrarlayamama eğilimi

ile karakterizedir. Ek olarak ürünler zamanında canlıya alınamamaktadır, maliyetlerde aşmalar ve teslim edilen kalite beklentilere uygun değildir.

6.2.2. Yönetilen: Seviye 2

TMMi seviye 2'de test, yönetilen bir süreç haline gelir ve hata ayıklamadan açıkça ayrılır. Olgunluk seviyesi 2'ye yansıyan süreç disiplini, stresin olduğu zamanlarda mevcut uygulamaların korunmasını sağlar. Test birçok paydaş tarafından kodlamayı takip eden bir proje aşaması olarak algılanmaktadır. Test sürecinin iyileştirilmesi bağlamında, şirket çapında veya program çapında bir test stratejisi oluşturulur. Test planında, yaklaşımın bir ürün risk değerlendirmesinin sonucuna dayandığı bir test yaklaşımı tanımlanmıştır. Belge gereksinimlerine göre ürün risklerini tanımlamak için risk yönetimi teknikleri kullanılır. Test planı, hangi testin ne zaman, ne zaman ve kimin tarafından yapıldığını tanımlar. Taahhütler paydaşlar ile oluşturulmakta ve gerektiğinde revize edilmektedir. Testin devam etmesini sağlamak için izlenir ve kontrol edilir. İş ürünlerinin durumu ve test hizmetlerinin sunulması yönetim tarafından görülebilir. Test tasarım teknikleri, spesifikasyonlardan test vakalarının türetilmesi ve seçilmesi için uygulanır. Test geliştirme yaşam döngüsünde geç başlayabilir, ör. Tasarım sırasında veya hatta kodlama aşamasında. TMMi Seviye 2'de birim, entegrasyon, sistem ve kabul testi seviyeleri vardır. Tanımlanan her test seviyesi için, kuruluş çapında veya program çapında test stratejisinde tanımlanan özel test hedefleri vardır. Test ve hata ayıklama işlemleri farklıdır. TMMi seviye 2 de test edilmenin temel amacı, ürünün belirtilen gereksinimleri karşıladığını doğrulamaktır. Bu TMMi seviyesindeki birçok kalite problemi, testin geliştirme yaşam döngüsünde geç gerçekleşmesi nedeniyle meydana gelir. Hatalar gereksinimlerden yayılır ve tasarıma kodlanır. Resmi bir inceleme bulunmamaktadır. Yürütme tabanlı test birçok paydaş tarafından birincil test aktivitesi olarak kabul edilir.

Test politikası ve stratejisi

Test politikası ve stratejisi süreç alanı, test politikasının ve stratejisinin uygulanması ve tanımını içerir. Test stratejisinde test seviyeleri belirlenir. Her test seviyesi için test hedefleri, sorumluluklar, temel görevler ve giriş / çıkış kriterleri tanımlanır. Test performansını ve test iyileştirme hedeflerinin başarısını ölçmek için, test performans göstergeleri tanımlanır ve uygulanır. Test politikası ve strateji süreç alanının amacı, test seviyelerinin kesin olarak

tanımlandığı bir test politikası ve kurum çapında veya program çapında bir test stratejisi geliştirmek ve oluşturmaktır.

Test Planlaması

Test planlaması, test objesi üzerinde bir ürün risk değerlendirmesi gerçekleştirilmesini ve tanımlanan risklere göre farklı bir test yaklaşımının tanımlanmasını içerir. Ayrıca, yapılacak testler için tahminlerin geliştirilmesini, gerekli taahhütlerin oluşturulmasını ve testin yönlendirilmesi ve yönetilmesi için planın tanımlanmasını ve sürdürülmesini içerir. Her bir test seviyesi için bir test planı gereklidir. TMMi 2 seviyesinde test planları tipik olarak test seviyesi başına geliştirilir. Test planlamasının amacı, tanımlanan risklere ve tanımlanmış test stratejisinin baz alındığı test yaklaşımının tanımlanması, test faaliyetlerinin yürütülmesi ve yönetilmesi için iyi kurulmuş planlar oluşturmak ve sürdürmektir.

Test İzleme ve Kontrol

Süreç izleme ve kontrol süreci, test ilerlemesini ve ürün kalitesini belgelendirilmiş tahminlere, taahhütlere, planlara ve beklentilere karşı izlemeyi, test ilerlemesini ve ürün kalitesini paydaşlara bildirmeyi, kontrol önlemlerini almayı (gerektiğinde düzeltici eylemleri) ve kapatmak için düzeltici eylemleri yönetmeyi içerir. Test izleme ve kontrol süreç alanının amacı, test ilerlemesinin ve ürün kalitesinin anlaşılmasını sağlamaktır. Böylece test ilerlemesi plan ve ürün kalitesinden belirgin bir şekilde saptığında ya da beklentilerden önemli ölçüde saptığında uygun düzeltici önlemlerin alınabilir.

Test Tasarımı ve Yürütülmesi

Test tasarımı ve uygulaması süreç alanı, test koşulları ve test durumlarını türetmek ve seçmek için test tasarım tekniklerinin uygulanmasını içeren test hazırlama aşamasını ele almaktadır. Ayrıca belirli test verilerinin oluşturulmasını, belgelenmiş test prosedürlerini ve olay yönetimini kullanarak testlerin yürütülmesini de ele alır. Test tasarım ve uygulamasının amacı, test tasarım özellikleri belirleyerek, test tasarım teknikleri kullanarak, yapılandırılmış bir test yürütme süreci gerçekleştirmesi ve test durumlarını kapatmayı yönetilmesi yoluyla, test tasarımı ve yürütme süresince test süreci yeteneğini geliştirmektir.

Test Ortamı

Test ortamı gereksinimlerini belirlemek, test ortamını uygulamak, test ortamını yönetmek ve kontrol etmek gibi aktivitelere hitap eder. Test ortamının yönetimi ve kontrolü ayrıca yapılandırma yönetimi ve kullanılabilirliği sağlama gibi hususları içerir. Test ortamı süreç alanının amacı, testlerin yönetilebilir ve tekrarlanabilir bir şekilde yürütülmesinin mümkün olduğu test verileri de dâhil olmak üzere uygun bir ortam oluşturmak ve sürdürmektir.

6.2.3. Tanımlı: Seviye 3

TMMi modeli tanımlı seviyesinde test kodlamanın takip edildiği bir süreç olmaktan çıkmıştır. Yazılım geliştirme yaşam döngüsüne ve ilgili dönüm noktalarına tamamen entegre edilmiştir. Test planlaması, gereksinimler sırasında ve bir ana test planında belgelenen, erken bir proje aşamasında yapılır. Kuruluşun olgunluk seviyesi 3ün temeli olan standart test süreçleri seti, zaman içinde oluşturulmakta ve geliştirilmektedir. Bir test organizasyonu ve özel bir test eğitim programı vardır ve test bir meslek olarak algılanır. Test süreci iyileştirme, test kuruluşunun kabul ettiği uygulamaların bir parçası olarak tamamen kurumsallaştırılmıştır. Seviye 3'teki kuruluşlar, gözden geçirmelerin kalite kontrolde önemini; Dinamik test sürecine henüz tam olarak bağlı olmamasına rağmen resmi bir gözden geçirme programı uygulanmaktadır. Test uzmanları, gereksinim özelliklerinin gözden geçirilmesinde yer alır. TMMi 2 seviyesindeki test tasarımları esas olarak fonksiyonel testleri, test tasarımları ve test teknikleri üzerinde odaklanarak iş hedeflerine bağlı olarak fonksiyonel olmayan test, örneğin kullanılabilirlik ve / veya güvenilirlik dâhil olmak üzere 3ncü seviyede genişletilir. İkinci olgunluk seviyesi ve üçüncü olgunluk seviyesi arasındaki önemli fark, standartların, süreç tanımlarının ve prosedürlerin kapsamıdır. TMMİ modeli 2nci olgunluk seviyesinde, örneğin belirli bir projede oldukça farklı olabilir. 3ncü olgunluk seviyesinde ise bunlar, belirli bir proje veya organizasyon birimine uymak için kuruluşun standart süreçler kümesinden uyarlanmıştır ve bu nedenle uyarlama kurallarının izin verdiği farklılıklar haricinde daha tutarlıdır. Bir başka kritik ayrım ise, olgunluk seviyesi 3 de, süreçlerin tipik olarak olgunluk seviyesi 2'den daha detaylı bir şekilde açıklanmasıdır. Sonuç olarak, olgunluk seviyesi 3'te, kuruluş olgunluk seviyesi 2 süreç alanlarını tekrar gözden geçirmelidir.

Test Organizasyonu

Test organizasyonu süreç alanı, işleyişi (görevler, sorumluluklar, raporlama yapısı) ve genel organizasyondaki bir test grubunun konumunu tanımlar. Test rolleri, fonksiyonlar ve kariyer yolları testin profesyonel bir disiplin olarak kabul edilmesini desteklemek için tanımlanmıştır. Test organizasyonunda test sürecinin iyileştirilmesi, mevcut test sürecinin değerlendirilmesini ve olası test iyileştirmelerini tanımlamak için öğrenilen dersleri, geliştirmeleri uygulamak ve projelerin test faaliyetlerinde bunları dağıtmaktır. Test organizasyonu süreç alanının amacı, testten sorumlu bir grup yetenekli insanı tanımlamak ve organize etmektir. Test grubuna ek olarak, test grubu kuruluşun test sürecine ve test süreci varlıklarına, kuruluşun mevcut test sürecinin ve test süreci varlıklarının güçlü ve zayıf yönlerini tam olarak anlamaya dayalı iyileştirmeleri de yönetir.

Test Eğitim Programı

Süreç alanı test eğitim programı, organizasyonel test eğitim planının ve test eğitim kabiliyetinin oluşturulmasına yöneliktir. Planlanan test eğitiminin gerçek teslimini de ele alır. Projeye özel eğitim ihtiyaçları bu süreç alanının bir parçası değildir. Bunlar test planlaması süreç alanında ele alınmaktadır. Test eğitim programı süreç alanının amacı, insanlara bilgi ve beceri geliştirmeyi kolaylaştıran bir eğitim programı geliştirmektir. Böylece test görevleri ve rolleri etkin ve verimli bir şekilde gerçekleştirilebilir.

Test Yaşam Döngüsü ve Entegrasyonu

Test yaşam döngüsü ve entegrasyon, kullanılabilir bir dizi organizasyonel test süreci varlığını (örneğin standart bir test yaşam döngüsü) ve çalışma ortamı standartlarını oluşturmak ve sürdürmek ve test yaşam döngüsünü geliştirme yaşam döngüsü ile entegre etmek ve senkronize etmek için tüm uygulamaları ele almaktadır. Entegre yaşam döngüsü bir projeye testin erken katılımını sağlar. Test yaşam döngüsü ve entegrasyonunun amacı, tanımlanmış olana göre birden fazla test seviyesinde tutarlı bir test yaklaşımı tanımlamaktır.

Fonksiyonel Olmayan Test

Fonksiyonel olmayan test süreç alanı, fonksiyonel olmayan bir ürün risk değerlendirmesi gerçekleştirmeyi ve tanımlanan fonksiyonel olmayan riskleri temel alan bir test yaklaşımı tanımlamayı içerir. Fonksiyonel olmayan test süreci alanının amacı test planlama, test tasarımı ve yürütme sırasında fonksiyonel olmayan testlerin dâhil edilmesi için test süreci yeteneğini geliştirmektir. Tanımlanmış fonksiyonel olmayan ürün risklerine dayanan bir test yaklaşımı tanımlayarak, fonksiyonel olmayan test spesifikasyonları belirleyerek ve fonksiyonel olmayan testlere odaklanmış yapılandırılmış bir test yürütme sürecini yürüterek yapılır.

Eş Gözden Geçirme

Eş gözden geçirme süreci alanının amacı, iş ürünlerinin belirtilen gereksinimleri karşıladığını ve seçilen iş ürünlerindeki hataları erken ve etkin bir şekilde ortadan kaldırdığını doğrulamaktır. Gözden geçirmeler, değişikliklerin gerekli olduğu alanları ve hataları tanımlamak için meslektaşları tarafından iş ürünlerinin yöntemsel olarak incelenmesini içerir. İncelemeler, genellikle 2-7 kişi arasında, küçük bir mühendis grubuyla yapılır. İncelenecek olan iş ürünü, bir şartname, tasarım belgesi, kaynak kodu, test tasarımı, kullanım kılavuzu veya başka bir belge türü olabilir. Pratikte hakem grubunun seçildiği birçok yol vardır. Yorum yapanlar şunlar olabilir:

- Gözden Geçirme alanında uzmanlar (kalite güvence veya denetim)
- Aynı projeden insanlar
- Yazarı tarafından belirli bilgileri nedeniyle davet edilen kişiler
- İnsanlar, ör. Ürüne özel ilgi gösteren işletme temsilcileri

Bir örnekte yazar bir grup insanı bir dokümana ve onun düşünce sürecine yönlendirir, bu sayede tüm çalışanlar belgeyi anlar ve içerik ya da yapılacak değişikliklerle ilgili bir fikir birliğine varır. Teknik bir incelemede grup bireysel bir hazırlıktan sonra kullanılacak içerik ve teknik yaklaşımı tartışır. En resmi inceleme türü olan bir inceleme, bir belgenin her birey ve grup tarafından kaynaklar ve standartlar kullanarak ve öngörülen kuralları izleyerek hatalar için kontrol edildiği bir tekniktir.

6.2.4. Ölçülen: Seviye 4

TMMi seviye 2 ve 3ün hedeflerine ulaşmak, test sürecinin iyileştirilmesi için kapsamlı bir test ve destek sağlayabilen teknik, yönetsel ve personel altyapısının devreye sokulmasının faydalarına sahiptir. TMMi 4ncü seviye organizasyonlarında, testler tamamen tanımlanmış, sağlam ve ölçülebilir bir süreçtir. Değerlendirme olarak algılanmaya başlayan test bu seviyede ürünleri kontrol etmekle ilgili tüm yaşam döngüsü faaliyetlerinden oluşur. Test sürecinin kalitesini değerlendirmek, verimliliği değerlendirmek ve iyileştirmeleri izlemek için kullanılabilecek kurum çapında bir test ölçüm programı devrededir. Bir test ölçüm programı ayrıca test performansına ve maliyetine ilişkin tahminleri de destekler. Ürün kalitesine ilişkin olarak bir ölçüm programının varlığı bir kuruluşun kalite ihtiyaçlarını, kalite özelliklerini ve kalite ölçümlerini tanımlayarak bir ürün kalitesi değerlendirme sürecini gerçekleştirmesine olanak tanır. Ürün kalitesi niceliksel olarak anlaşılır ve yaşam döngüsü boyunca tanımlanmış hedeflere göre yönetilir. İncelemeler ve gözden geçirmeler, test sürecinin bir parçası olarak kabul edilir ve yaşam döngüsünün başlarında ürün kalitesini ölçmek ve kaliteyi kontrol etmek için kullanılır. Bir hata tespit tekniği olarak eş değer değerlendirmeleri, ürün kalite değerlendirme sürecine uygun olarak ürün kalite ölçüm tekniğine dönüştürülür. TMMi seviye 4, eş gözden geçirmeleri (statik test), dinamik testler ve daha etkili bir şekilde test yapmayı amaçlayan test yaklaşımını optimize etmek için eş gözden geçirme sonuçlarının ve verilerin kullanımı arasında koordineli bir test yaklaşımı oluşturmayı kapsar.

Test ölçümü süreç alanı

Test ölçümü süreç alanında test sürecinin etkinliğini arttıracak ölçütler toplanması, analiz edilmesi ve uygulanması amaçlanmıştır. Test ölçümü süreç alanı, yazılım ürünlerinde ve test işleyişinde adım adım anlaşılması, yazılım geliştirme sürecinin etkili olmasına destek olması için veri tanımlamanın, veri toplamanın ve bulunan bu verilerin analiz edilmesinin kapsandığı sürekliliği sağlanan ve bu şekilde devam edilmesi kontrol edilebilen bir süreçtir. Test ölçüm sürecinde uygulamaların desteklenmesi için veri toplanması, depolanması, yeniden kullanılması ve aktarımı için ölçüt ve analiz metotları özelleştirilir. Test ölçüm sürecinde iki temel alan bulunur. Bu alanlar ile ürün kalitesinin geliştirilmesi ve süreç geliştirmesi konularına katkılar sağlanır. Bu durumların adım adım başarı sağlayabilmesi için test ölçüm programı organizasyonların amaçlarıyla, test politikasıyla ve test stratejisiyle

uyumlu ve bağlantılı bir şekilde bulunması gereklidir. Yazılım sektöründe bulunan şirketlere ait organizasyonların hedefledikleri durumlar, test ölçüm hedeflerinin ve metriklerinin oluşmasında ilk adım olarak kabul edilir. Bu adımlar başarılı bir şekilde uygulanırsa test ölçüm programı şirketlerin hedefleriyle uyumlu hale gelecektir ve ölçütler bütün test çalışanları için kabul edilebilir durumda olacaktır. Verilerin toplanması ve analiz edilmesi sonucu belirlenen ölçütler sonraki gerçekleştirilmesi planlanan projeler için organizasyonun plan geliştirmesine katkıda bulunacaktır. Belirlenen bu test ölçütlerine ve metriklerine test maliyeti, test senaryolarının sayısı, bulunan hata ile ilgili bilgiler, test eforu, gereksinimlerin sayısı ve yazılım ürün ölçütleri örnek olarak gösterilebilir.
Test ölçüm süreç alanı aşağıdakileri içerir.

- Belirlenecek gereksinimler ve şirket amaçlarının da bulunduğu test ölçütlerinin amaçlarının birlikte bir uyum halinde olması

- Verilerin toplanması, depolanması, tekrar kullanılması ve geri besleme fonksiyonlarının belirtilmesi

- Verilerin toplanması ve analiz edilmesi sonrası elde edilen ölçütlerin ve analizlerin tekniklerinin belirtilmesi

- Ölçütlerde ve analizde bulunan tekniklerin doğru şekilde kullanılması ve doğru kararların verilebilmesi için kullanılacak olan sonuçların sağlanması.

Ürün kalite değerlendirmesi süreç alanı

Ürün kalite değerlendirmesi süreç alanında amaç yazılım ürünün kalitesiyle ilgili nicel bir öngörülebilirlik sağlanması ve bu sayede yazılım geliştirme projelerinin istenen ve beklenen ürün kalitesine ulaşmaktır. Ürün kalite değerlendirme süreç alanı, projenin nicel ürün kalite hedef tanımlarındaki yapılacak ya da yapılması planlanan aktivitelerini ve ürün kalite değerlendirmesinde kullanılacak kalite ölçümlerini içerir. Bu süreç alanında genel olarak hedeflenen, müşteriler ya da kullanıcılar için kaliteli ve etkin yazılım ürünlerin ortaya çıkarılması ve müşterilerin gereksinimlerinin karşılanabilmesidir. Bu süreç alanında nicel olarak belirlenen hedefler, şirketlerin organizasyonlarının, müşterinin ve kullanıcının gereksinimlerini karşılayan yazılım ürünlerinin oluşması için bulunur. Bu hedeflerin başarı sağlaması için organizasyonların kendi test süreçleri ile nicel hedeflerin uyumlu olması beklenir ve şirketler organizasyonlarında bu süreç alanında kalitede hedefledikleri duruma erişebilmek için stratejiler ve planlar geliştirir.

Gelişmiş gözden geçirmeler süreç alanı

Gelişmiş gözden geçirmeler süreç alanın amacı; TMMi üçüncü seviyedeki süreç alanındaki dinamik test ile eş gözden geçirmelerin temel olarak ele alınması ile test stratejilerinin ve test yaklaşımlarının geliştirilmesi ve yazılım geliştirme sonucunda elde edilen ürün kalitesinin ölçümünü yazılım geliştirme yaşam döngüsünün ilk safhalarında gerçekleştirebilmektir. Test sürecinin net ve doğru bir şekilde sonuçlanması, test sürecinin yazılım geliştirme yaşam döngüsü sonucu elde edilen yazılım ürünün ve ilgili iş ürünlerinin planlanması, hazırlanması ve değerlendirilmesinin istenen şekilde yapılmasıdır. TMMi modeli dördüncü olgunluk seviyesinde test sürecinin açık şekilde tamamlanmasını statik ve dinamik test ile desteklenir. TMMi üçüncü seviyedeki eş gözden geçirmeler süreç alanında eş gözden geçirmelerin yapılmakta fakat dinamik testin yapılmamasından dolayı bütünleştirilmemektedir. Eş gözden geçirmeler, ürün oluşturulurken hataların ve ürün risklerinin tanımlanması için etkilidir. Eş gözden geçirmeler ve dinamik test koordine edildiği zaman erken gözden geçirme sonuçları ve veriler test yaklaşımına yön vermek için kullanılabilir. Test süreci hataları gruplandırma prensibi üzerine oluşturuluşa gözden geçirmeler sırasında hataların tipi ve sayısının bulunması daha etkili test yaklaşımlarının geliştirilmesine katkıda bulunacaktır. Projenin ilerleyen periyotlarında test yaklaşımı tekrar gözden geçirilerek değerlendirilir ve güncellemeleri yapılabilir. Gözden geçirme sonucu

elde edilen bu veriler projenin her aşamasında yapılabilecek bu güncellemeler sırasında ele alınması gereken verilerden biridir.

6.2.5. Optimize: Seviye 5

TMMi modelinin 1'den 4'e kadar olan tüm önceki test iyileştirme hedeflerinin gerçekleştirilmesi tamamen tanımlanmış ve ölçülen bir süreci destekleyen test için bir organizasyon altyapısı yaratmıştır. Bir kuruluş, TMMi olgunluk seviyesinde istatistiksel olarak kontrol edilen süreçlerin niceliksel anlayışa dayalı olarak süreçlerini sürekli olarak geliştirebilir. Test süreci performansının iyileştirilmesi artan ve yenilikçi süreç ve teknolojik iyileştirmelerle gerçekleştirilmektedir. Test yöntemleri ve teknikleri optimize edilmiştir ve süreç iyileştirmeye sürekli odaklanmaktadır.

- Yönetilen, tanımlanmış, ölçülü, verimli ve etkili
- Sektörden kuruluşa teknoloji transferini destekleyebilir
- Test varlıklarının yeniden kullanımını destekleyebilir
- Hata önleme odaklı
- Otomasyonun desteklediği kadar kaynakların etkili bir şekilde kullanılması
- Sürekli iyileştirme elde etmek için süreç değişikliğine odaklanma
- İstatistiksel olarak kontrol edilebilir ve tahmin edilebilir

Test süreci altyapısının sürekli iyileştirilmesini desteklemek ve test iyileştirmelerini belirlemek, planlamak ve uygulamak için kalıcı bir test süreci iyileştirme grubu resmi olarak kurulur ve becerilerini arttırmak için özel eğitim almış üyeler tarafından ve grubun başarısı için gerekli bilgi verilir. Birçok organizasyonda bu gruba test süreci grubu denir. Bu grup için destek, test organizasyonu başlatıldığında üçüncü seviyede resmen başlar. Dördüncü ve beşinci seviyede, sorumluluklar daha yüksek seviyeli uygulamalar ortaya çıktıkça, örneğin yeniden kullanılabilir test varlıklarının tanımlanması ve test varlık kütüphanesinin geliştirilmesi ve sürdürülmesi olarak büyür. Hata önleme süreci alanı, geliştirme yaşam döngüsündeki yaygın kusurların nedenlerini tanımlamak ve analiz etmek ve benzer hataların gelecekte ortaya çıkmasını önlemek için eylemler tanımlamak üzere kurulmuştur. Süreç

alanları ve uygulamaları uygulama bölümünde ayrı ayrı olarak ele alınmış ve bir proje üzerinde gerçekleştirilmiştir.

Hataların önlenmesi süreç alanı

Hataların önlemesi süreç alanında, yazılım geliştirme yaşam döngüsünün gelişiminde hataların ortak ve kök nedeninin analiz edilmesi, tanımlanması ve sonraki aşamalarda çıkabilecek aynı tipteki hataların tanımlanması amaçlanmıştır. Beşinci olgunluk seviyesinde hata bulmaya yönelik süreç olmaktan çıkmıştır. Bu seviyede hataların önlenmesine yönelik bir durum söz konusudur. Bu doğrultuda test yapılırken hatalardan korunma sürecine odaklanılır. Hataların önlenmesi işlemi daha önce bulunan hataların tanımlama nedenlerini ve daha sonraki bulunacak hata çeşitlerinin oluşmasını önleyecek belirli görevleri yapmayı içerir. Analiz edilecek hataların seçimi, risk faktörü dâhil edilerek daha fazla çeşitli etkenlere dayanarak hatadan korunma alanlarına ve hataların en kritik olduğu alanlara odaklanmak gerekir. Hataların test sırasında bulunmadan önce önlenmesi genel olarak projede çıkacak fazla maliyeti engelleyecektir. Beşinci olgunluk seviyesinde şirketler, test sırasında hangi tip hatalardan korunması gerektiğini ve hangi tip hataların önlenmesi gerektiğini bileceğinden daha hatasıza yakın ürün oluşturabilecektir. Günlük yapılacak analizler için metot örnekleri, belirlenmiş günlük analiz toplantıları, hata analizinde kullanılacak araçların kullanımı ve neden sonuç ilişkileri, resmi incelemeler sırasındaki günlük yapılacak analiz ve standart hata gruplandırılması kullanımıdır. Hataların önlenmesi süreç alanı, hataların ortak nedenlerini tanımlayarak ve analiz ederek ve daha sonraki projelerde karşılaşılacak hata çeşitlerinin ortak nedenlerini yok ederek belirli aktiviteleri tanımlamayı projede ve organizasyonun herhangi bir yerinde adres gösterir. Yazılım geliştirme sırasında ve test etme safhalarında bulunan tüm hatalar bu süreç alanının içerisindedir.

Kalite kontrol süreç alanı

Kalite kontrol süreç alanında istatistiksel veri olarak test sürecini yönetilmesi ve kontrol edilmesi amaçlanmıştır. Bu seviyede test süreç performansı öngörülebilir ve kabul edilir sınırlarla ayarlanarak dengede tutulur. Yazılım geliştirme sonucunda elde edilen ürün kalitesi ile ilgili önceden fikir edinebilmek ve test sürecini daha etkili yürütebilmek için örnekler referans alınarak ve istatistiksel metotları kullanarak proje seviyesinde uygulanır. Kalite kontrol süreç alanındaki uygulamalar yazılım sonucu elde edilen ürünün standartlara

ve ihtiyaçlara uygun hale getirilmesini sağlamak için kullanılan süreç ve pratikleri içerir. Genel olarak kabul edilen her adım sonra oluşan çıktıya, yürütülmenin nasıl yapılacağını tarifleyen kurallar ve prosedürler kullanılarak karar verilir. Süreç adımını etkileyen birden fazla öngörülemeyen varyasyonlar oluştuğu durumda, süreç değişken bir hale gelir, tahmin edilebilirliği güçleşir ve kontrol edilemez. Tahmin edilemeyen ya da öngörülemeyen süreç haline geldiği durumda bu sürece kaliteli sonuçlar vermesi beklenmez.
Süreçlerin ölçülü bir şekilde kontrol edildiği bir organizasyonda aşağıdakiler yapılabilir:

- Yazılım test sürecinin standartlığına karar verilir.
- Tahmin edilemeyen ve öngörülemeyen süreçler tanımlanarak süreç performansı ile ilgili sınırlar tanımlanabilir.
- Organizasyonların mevcut süreçlerindeki gelişim olanakları tanımlanır.

Kalite kontrol süreci test yaşam döngüsü organizasyonlarda istenen ya da hedeflenen ve tanımlanmış halde bulunan performansları için hedefler oluşturmayı kapsamaktadır. Bu hedefler ise şirketlerin organizasyonları için tanımlanan test politikasını temel almaktadır.

Test süreç iyileştirmesi süreç alanı

Test süreç iyileştirme süreç alanında, şirketlerin test organizasyonunda kullanılan, mevcuttaki test süreçlerini sürekli olarak geliştirilmesi ve test teknolojilerinin uygun olacak şekilde tanımlanması ve uyumlu olarak organizasyona geçişinin sağlanması amaçlanmıştır. Bu süreç alanında bulunan bu amaçlar ve gelişmeler, şirketlerin organizasyonlarının test süreçleri amaçlarını kapsayacak şekilde yazılım geliştirme sonucu elde edilecek ürün kalitesi ve test süreç performans amaçlarını destekler. TMMi modelinin beşinci olgunluk seviyesinde, test süreci mevcutta devam eden ya da yapılacak projelerde ve tüm organizasyonu kapsayacak şekilde sürekli olarak gelişmelidir. Test sürecinin kapasite olarak büyümesinin devam eden ve süreklilik sağlayan bir süreç olması için ölçülebilirdir. Şirketler organizasyonlarında bu sağlanacak altyapının sürekli olarak gelişmesini desteklemek zorundadır. Bu alt yapı genel olarak politikalar, standartlar, eğitim faaliyetleri, araçlar ve organizasyonel yapılar içerir.

Test süreci iyileştirme süreç alanı genel olarak aşağıdakilerden oluşur:

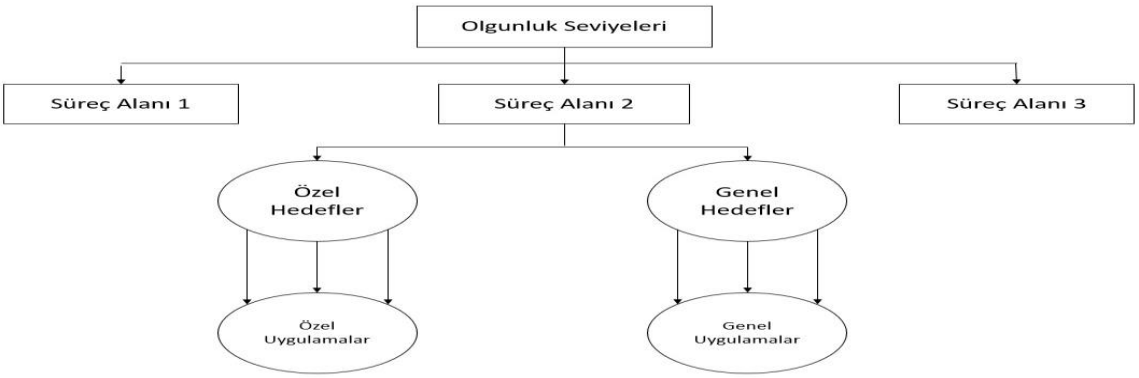
- Test süreç değerlendirme ve iyileştirme prosedürleri belirlenmesi
- Bu prosedürler ve değerlendirmeler için yetkililerin belirlenmesi
- Test uygulamalarında bulunan zayıf ve güçlü alanların tekrar kullanım için tanımlanması
- Test süreç ve teknolojilerini geliştiren uygulamaların organizasyon içinde devreye alınması
- Organizasyonda kullanılabilecek en faydalı uygulamaların belirlenmesi
- Gelişen ve değişen teknolojik çevreye uyum sağlayabilmek adına yeni test araçlarını ve teknolojilerin takip edilmesi
- Teknoloji ve bilgi aktarımının desteklenmesi ile kalitesi fazla olan test özelliklerinin tekrarlı bir şekilde kullanılması

Test sürecinin sürekli olarak geliştirilmesinin sağlanması, organizasyonun kendi mevcut test sürecini ve projelerde tanımlanan süreçlerinin çıkacak yeni ve değişen süreçlere uyumlu hale gelmesini sağlar. Test süreç iyileştirme ve gelişim eylemlerinde bulunacak çalışanlar için eğitim ve teşvik edici programlar oluşturulabilir. Test süreç iyileştirmesi süreç alanı yeni test teknolojileri seçilmesi ve bunları organizasyonda bulunan mevcut test sürecine entegre eden uygulamaları ve planlama, geliştirme, izleme, değerlendirme ve test gelişim hareketlerini ölçmeyi içerir.

6.3. Süreç Alanları

Seviye 1 haricinde belirtildiği üzere, her bir olgunluk seviyesi, bir kuruluşun test sürecini iyileştirmeye odaklanması gerektiğini gösteren çeşitli süreç alanlarından oluşur. Süreç alanları, olgunluk seviyesine ulaşmak için ele alınması gereken konuları tanımlar. Her süreç alanı, testle ilgili faaliyetlerin bir kümesini tanımlar. Uygulamaların tamamı gerçekleştirildiğinde, o alan ile ilgili faaliyetlerde önemli bir iyileştirme yapılacaktır. TMMi ‘de, sadece test süreci yeteneğinin temel belirleyicisi olarak kabul edilen süreç alanları tanımlanmıştır. Olgunluk seviyesi ve daha düşük olgunluk seviyelerinin tüm süreç alanları, elde edilecek bir olgunluk seviyesini dikkate almaktan memnun olmalıdır. Örneğin, bir

kuruluş TMMi seviye 3'te ise, TMMi seviye 2 ve TMMi seviye 3'teki tüm süreç alanlarını karşılamıştır.



Şekil.6.1. TMMi yapısı ve bileşenleri

6.4. TMMi Bileşenleri

6.4.1. Özel hedefler

Özel bir hedef, süreç alanının iyileştirilmesi için mevcut olması gereken benzersiz özelliği tanımlar. Özel bir hedef, gerekli bir model bileşenidir ve bir süreç alanının iyileştirildiğinin belirlenmesine yardımcı olmak için değerlendirmelerde kullanılır.

6.4.2. Genel hedefler

Genel hedef bir süreç alanının sonuna yakın görünür ve aynı hedef ifadesi birden çok süreç alanında göründüğü için genel olarak adlandırılır. Genel bir hedef, bir süreç alanını uygulayan süreçleri kurumsallaştırmak için bulunması gereken özellikleri açıklar. Genel bir hedef, gerekli bir model bileşenidir ve bir süreç alanının iyileştirildiğinin belirlenmesine yardımcı olmak için değerlendirmelerde kullanılır.

6.4.3. Özel uygulamalar

Özel bir uygulama, ilgili özel amaca ulaşmak için önemli kabul edilen bir aktivitenin tarifidir. Özel uygulama, bir süreç alanının belirli hedeflerine ulaşılması ile sonuçlanması beklenen faaliyetleri açıklar. Belirli bir uygulama beklenen bir model bileşenidir.

6.4.4. Genel uygulamalar

Aynı uygulama birden çok işlem alanında göründüğü için genel olarak adlandırılır. Genel bir uygulama, ilişkili genel hedefe ulaşmada önemli görülen bir etkinliğin açıklamasıdır. Genel bir uygulama beklenen bir model bileşenidir.

6.4.5. Örnek iş ürünleri

Örnek iş ürünleri bölümü, belirli bir uygulamadan örnek çıktıları listeler. Bu örneklere örnek iş ürünleri denir çünkü çoğu zaman etkili olan ancak listelenmeyen iş ürünleri vardır. Örnek bir iş ürünü, bilgilendirici bir model bileşenidir.

7. OLGUNLUK SEVİYE BELİRLEME VE DEĞERLENDİRME İÇİN BİR YÖNTEM ve UYGULAMA

Bu bölümde test süreçlerinde iyileştirme çalışmalarına odaklanan yazılım sektöründe faaliyet gösteren firmalardan TMMi 2nci ve 3üncü seviye değerlendirmelerini, değerlendirmeden önce kendi içlerinde yapabilmelerine yönelik geliştirilen olgunluk seviye belirleme ve değerlendirme yöntemi verilmiştir.

7.1. Yöntem

Yazılım test süreçlerinde iyileştirilmenin sağlanması, olgunluk seviyesinin artırılması uluslararası platformlarda da geçerliliğinin sağlanması için detaylı bir model olan TMMi modelinin bir firmada gerçekten uygulanabilmesi fazlasıyla emek ve zaman gerektiren ayrıca uygulama sonucunda istenilen hedefe ulaşıp bunu sertifika ile belgelendirmek şirketler için ek bir maliyet olarak düşünülebilecek bir süreçtir. TMMi değerlendirmesi yaptırmak isteyen firmaların bir kısmı resmi bir sertifikasyon sahibi olmak için isterken bir kısmı da sadece organizasyondaki test süreçlerinde iyileştirme yapmak için isterler. TMMi sertifikası için TMMi vakfı tarafından lider değerlendiriciler ile başvuru yapan firmalarda geniş bir analiz yapılır, süreç iyileştirme çalışmaları bütün çalışanlara aktarılır, ikna edilir ve bu hazırlıklar tamamlandıktan sonra süreç iyileştirme çalışmalarına geçilir. Bu süre içerisinde mevcutta devam edecek işlerinde olması çalışanlarca ekstra efor ve zaman kaybı olarak görüleceğinden fazlasıyla emek gerektiren bir süreç halini alır. Bu durumlardan kaçınmak adına ve ek maliyetlere girmek istemeyen ve dolayısıyla yazılım ürünlerindeki hatalardan kaynaklanan maliyetlerden katlanmak durumunda kalan ve test süreçlerinde iyileştirme yapamayan firmalara yardımcı olmak adına bir yöntem geliştirilmiştir. Bu yöntem yazılım sektöründe faaliyet gösteren ve test süreçlerinde iyileştirme yapmak isteyen telekomünikasyon alanında boy gösteren şirketin geliştirmekte olduğu bir proje üzerinde uygulanmıştır.

Yöntem, TMMi 2nci ve 3ncü olgunluk seviyeleri kapsamında Seviye 2 düzeyi için 10 ve Seviye 3 düzeyi için 10 ifadeden oluşmaktadır. Aynı formatta hazırlanan bu ifadeler 5'li likert ölçeğine göre derecelendirilmiştir. Buna göre verilecek cevapların sayısal karşılığı aşağıdaki gibidir:

- 5: Kesinlikle Evet
- 4: Evet
- 3: Kararsızım
- 2: Hayır
- 1: Kesinlikle Hayır

Seviye 2 ve Seviye 3 için oluşturulmuş yöntemde bulunan ifadeler aşağıdaki tabloda verilmiştir.

Çizelge 7.1. Seviye 2 değerlendirme ölçümü

İfadeler	Kesinlikle Evet	Evet	Kararsızım	Hayır	Kesinlikle Hayır
1.Şirketinizde projeler ya da talepler için test politikası ve test stratejileri tanımlanıyor.					
2.Test performans göstergeleri(test efor ve maliyeti, test süresi, bulunan hata sayısı, hata tespit yüzdesi, test kapsamı vb.) tanımlanıyor.					
3.Test planlaması ve risk değerlendirmesi yapılıyor.					
4.Test yaklaşımları (test edilecek öğeler, giriş ve çıkış kriterleri, durdurma ve başlatma kriterleri vb.) tanımlanıyor.					
5.Plana dayalı test ilerleme süreci izleniyor ve raporlamalar yapılıyor.					
6.Her test seviyesi(Birim, Entegrasyon, Sistem, Kabul testleri) açık bir şekilde tanımlanarak test hedefleri oluşturuluyor.					
7.Test tasarım teknikleri(Kara kutu, Beyaz kutu, Deneyim temelli test teknikleri vb.) kullanılıyor.					
8.Test durumları tanımlanıyor ve öncelik belirleniyor.					
9.Hata yönetim süreci bulunuyor, test sırasında karşılaşılan hatalar raporlanıyor.					
10.Yönetilebilir ve kontrol edilebilir test ortamı bulunuyor.					

Çizelge 7.2. Seviye 3 değerlendirme ölçümü

İfadeler	Kesinlikle Evet	Evet	Kararsızım	Hayır	Kesinlikle Hayır
1.Tanımlanmış test organizasyonu, test rolleri ve sorumlulukları bulunuyor.					
2.Test süreç iyileştirme planlamaları bulunuyor.					
3.Yazılım test alanında eğitim imkânları sağlanıyor ve eğitim planları uzmanlık seviyelerine göre oluşturuluyor.					
4.Test seviyelerine hitap eden tanımlanmış test yaşam döngüleri bulunuyor.					
5.Tanımlanmış test süreç veritabanı ve kütüphanesi bulunuyor.					
6.Şirketinizde yazılım geliştirme yaşam döngülerine entegre edilmiş test süreci bulunuyor.					
7.Fonksiyonel olmayan testler için resmi test yaklaşımları (test edilecek öğeler, giriş ve çıkış kriteleri vb.) bulunuyor.					
8.Fonksiyonel olmayan testler (Stres, Yük, Güvenlik, Performans, Kullanılabilirlik vb.) gerçekleştiriliyor.					
9.Kurulmuş bir gözden geçirme süreci yaklaşımı bulunuyor.					
10.Statik test teknikleri olarak gözden geçirme süreçleri işletiliyor ve statik test analizleri yapılıyor.					

Hazırlanan bu yöntemdeki seviyelerde verilen ifadeler ISTQB standartları, yazılım test süreçleri ve TMMi model çerçevesi içerisinde verilen süreç alanları ve bu süreç alanları için verilen özel uygulamalar kapsayacak şekilde oluşturulmuştur. Bu yöntem hazırlanırken literatür taraması yapılarak daha önceki çalışmalardan da ilham alınmıştır. Test olgunluk seviyesi belirleme ve değerlendirme için TMMi modeli ikinci ve üçüncü olgunluk seviyesinde bulunan tüm konuları dikkate alınarak ifadeler hazırlanmıştır. Hazırlanan bu yöntemdeki seviyelerde verilen ifadelerin genel anlamda hangi konuları kapsadığı ve amacın ne olduğu aşağıda sırayla verilmiştir.

Seviye 2 için verilmiş olan ifadeler: Bu seviyede şirketlerde tanımlanmış bir test süreci olup olmadığını, testin yazılım geliştirme aşamasından bağımsız mı ilerlediğini, test izleme ve kontrolünün test planına göre mi yapıldığını sorgulayan ifadeler bulunmaktadır. Ayrıca risk yönetiminin gereksinimlere bağlı şekilde mi yapıldığını, test stratejisi, test planları, test durumlarının teste başlamadan mı hazırlandığını anlamak amacını sorgulayan ifadeler bulunmaktadır.

Seviye 3 için verilmiş ifadeler: Bu seviyede şirketlerde test, yazılım geliştirme süreçlerine entegre olup olmadığını, tanımlanmış bir test organizasyonu ve test eğitimlerinin olup olmadığını, test süreci iyileştirme çalışmalarının şirketin hedefleri arasında olup olmadığını, şirket içerisinde testin meslek olarak algılanıp algılanmadığını, gereksinimlerin gözden geçirilmesinde test uzmanlarının bulunup bulunmadığını anlamak amacını sorgulayan ifadeler bulunmaktadır. Sonraki bölümde 2nci ve 3ncü seviyelerde bulunan süreç alanları ve bu süreç alanlarında bulunan özel uygulamaları tablolarda verilmiştir.

7.2. Seviye 2 Süreç Alanları

Seviye 2 düzeyinde bulunan ve uygulamaları bir sonraki bölümde gerçekleştirilen süreç alanları ve özel hedef ile özel uygulamalar aşağıdaki tablolarda verilmiştir.

Çizelge 7.3. Test politikası ve stratejisi süreç alanı özel hedef ve uygulamaları

Özel Hedefler (SG)	**Özel Uygulamalar (SP)**
SG 1 Bir Test Politikası Belirleme	SP 1.1 Test hedeflerini tanımlama
	SP 1.2 Test politikasını tanımlama
	SP 1.3 Test politikasını ilgili kişilerle paylaşma
SG 2 Test Stratejisi Belirleme	SP 2.1 Genel bir ürün risk değerlendirmesi yapılması
	SP 2.2 Test stratejisi tanımlama
	SP 2.3 Test stratejisini ilgili kişilerle paylaşma
SG 3 Test Performans Göstergelerini Belirleme	SP 3.1 Test performans göstergelerini tanımlama
	SP 3.2 Test performans göstergelerini uygulama

Çizelge 7.4. Test planlaması süreç alanı özel hedef ve uygulamaları

Özel Hedefler (SG)	**Özel Uygulamalar (SP)**
SG 1 Ürün Risk Değerlendirmesi	SP 1.1 Ürün risk kategorileri ve parametreleri
	SP 1.2 Ürün riskleri
	SP 1.3 Ürün riskleri analizi
SG 2 Bir Test Yaklaşımının Kurulması	SP 2.1 Test edilecek öğeleri ve özellikleri tanımlama
	SP 2.2 Test yaklaşımını tanımlama
	SP 2.3 Giriş kriterlerini tanımlama
	SP 2.4 Çıkış kriterlerini tanımlama
	SP 2.5 Durdurma ve yeniden başlatma kriterlerini tanımlama
SG 3 Test Tahminlerini Oluşturma	SP 3.1 Üst düzey bir iş analizi yapısı kurma
	SP 3.2 Test döngüsü tanımlama
	SP 3.3 Test efor ve maliyet için tahminleri belirleme
SG 4 Test Planı Geliştirilmesi	SP 4.1 Test programını belirleme
	SP 4.2 Test personelini planlama
	SP 4.3 Paydaş katılımını planlama
	SP 4.4 Test projesi risklerini tanımlama
	SP 4.5 Test planının oluşturulması

Çizelge 7.5. Test izleme ve kontrol süreç alanı özel hedef ve uygulamaları

Özel Hedefler (SG)	**Özel Uygulamalar (SP)**
SG 1 Plana dayalı Test İlerleme Sürecini İzleme	SP 1.1 Test planlama parametrelerini izleme
	SP 1.2 Sağlanan ve kullanılan test ortamı kaynaklarını izleme
	SP 1.3 Test projesi risklerini izleme
	SP 1.4 Test ilerlemesi gözden geçirmeleri
SG 2 Plan ve beklentilere göre Ürün Kalitesini İzleme	SP 2.1 Giriş kriterlerine göre kontrol
	SP 2.2 İzleme Hataları
	SP 2.3 Ürün risklerini izleme
	SP 2.4 Çıkış kriterlerini izleme
	SP 2.5 Durdurma ve tekrar başlama kriterlerini izleme
	SP 2.6 Ürün kalitesi gözden geçirmeleri
SG 3 Kapanışı Düzeltici Faaliyetleri Yönetin	SP 3.1 Analiz sorunları
	SP 3.2 Düzeltici eylemi gerçekleştirin
	SP 3.3 Düzeltici eylemi yönetin

Çizelge 7.6 Test tasarımı ve yürütülmesi süreç alanı özel hedef ve uygulamaları

Özel Hedefler (SG)	**Özel Uygulamalar (SP)**
SG 1 Test Tasarım Tekniklerini Kullanarak Test Analizi ve Tasarımı Yapılması	SP 1.1 Test koşullarının tanımlanması ve önceliklendirilmesi
	SP 1.2 Test durumlarının tanımlanması ve öncelik belirlenmesi
	SP 1.3 Spesifik test verilerinin tanımlanması
	SP 1.4 Gereksinimlerle yatay izlenebilirliğin korunması
SG 2 Test Uygulamasının Gerçekleştirilmesi	SP 2.1 Test prosedürlerinin geliştirilmesi ve önceliklendirilmesi
	SP 2.2 Özel test verilerinin oluşturulması
	SP 2.3 Giriş testi prosedürünün belirtilmesi
	SP 2.4 Test yürütme programının geliştirilmesi
SG 3 Test Yürütülmesinin Gerçekleştirilmesi	SP 3.1 Giriş testi yapılması
	SP 3.2 Test durumlarının yürütülmesi
	SP 3.3 Test Durum Raporları
	SP 3.4 Test loglarının Yazılması
SG 4 Test Durumları Kapanışının Yönetilmesi	SP 4.1 Yapılandırma kontrol panosunda test olaylarının düzenlenmesi
	SP 4.2 Test durumlarında karşılaşılan hataları düzeltmek için uygun eylemin gerçekleştirilmesi
	SP 4.3 Test hata durumunun izlenmesi

Çizelge 7.7. Test ortamı süreç alanı özel hedefleri ve özel uygulamaları

Özel Hedefler (SG)	**Özel Uygulamalar (SP)**
SG 1 Test Ortamı Gereksinimlerinin Geliştirilmesi	SP 1.1 Test ortamı ihtiyacı oluşması
	SP 1.2 Test ortamı gereksinimlerinin geliştirilmesi
	SP 1.3 Test ortamı gereksinimlerinin analizi
SG 2 Test Ortamı Uygulamasının Gerçekleştirilmesi	SP 2.1 Test ortamının uygulanması
	SP 2.2 Genel test verileri oluşturulması
	SP 2.3 Test ortamı giriş testi prosedürünün belirtilmesi
	SP 2.4 Test ortamı giriş testinin gerçekleştirilmesi
SG 3 Test Ortamlarının Yönetilmesi ve Kontrol Edilmesi	SP 3.1 Sistem yönetiminin gerçekleştirilmesi
	SP 3.2 Test veri yönetiminin gerçekleştirilmesi
	SP 3.3 Test ortamlarının kullanılabilirliğinin ve kullanımının koordine edilmesi
	SP 3.4 Test ortamı olaylarının rapor edilmesi ve yönetilmesi

7.3. Seviye 3 Süreç Alanları

Seviye 3 düzeyinde bulunan ve uygulamaları bir sonraki bölümde gerçekleştirilen süreç alanları ve özel hedef ile özel uygulamalar aşağıdaki tablolarda verilmiştir.

Çizelge 7.8. Test organizasyonu süreç alanı özel hedef ve uygulamaları

Özel Hedefler (SG)	**Özel Uygulamalar (SP)**
SG 1 Test Organizasyonu Kurma	SP 1.1 Test organizasyonu tanımlama
SG 2 Test Uzmanları için Test Fonksiyonları Kurma	SP 2.1 Test fonksiyonlarını tanımla
	SP 2.2 İş tanımlarını geliştirme
	SP 2.3 Personeli test fonksiyonlarına atama
SG 3 Test Süreci İyileştirmelerini Belirleme, Planlama ve Uygulama	SP 3.1 Kuruluşun test sürecini değerlendirme
	SP 3.2 Kuruluşun test süreci iyileştirmelerini tanımlama
	SP 3.3 Test süreci iyileştirme planlaması
	SP 3.4 Test süreci iyileştirmelerini uygulama

Çizelge 7.9. Test eğitim programı süreç alanı özel hedef ve uygulamaları

Özel Hedefler (SG)	**Özel Uygulamalar (SP)**
SG 1 Örgütsel Test Eğitimi Yeteneği Kurmak	SP 1.1 Stratejik test eğitimi ihtiyaçlarını belirleme
	SP 1.2 Organizasyon ve proje testi eğitim ihtiyaçlarını uyumlu hale getirme
	SP 1.3 Organizasyonel test eğitim planı oluşturulması
	SP 1.4 Test eğitim kabiliyetinin oluşturulması
SG 2 Test Eğitimi Sağlama	SP 2.1 Test eğitimi verilmesi
	SP 2.2 Test eğitim kayıtlarının oluşturulması

Çizelge 7.10. Test yaşam döngüsü ve entegrasyonu özel hedef ve uygulamaları

Özel Hedefler (SG)	**Özel Uygulamalar (SP)**
SG 1 Örgütsel Test Süreci Varlıklarını Kurma	SP 1.1 Standart test süreçleri kurmak
	SP 1.2 Tüm test seviyelerine hitap eden test yaşam döngüsü model açıklamaları oluşturmak
	SP 1.3 Uygunluk kriterlerini ve kurallarını oluşturmak
	SP 1.4 Kuruluşun test süreci veri tabanını kurma
	SP 1.5 Kuruluşun test süreci varlık kütüphanesini kurma
	SP 1.6 Çalışma ortamı standartlarını belirleme
SG 2 Test Yaşam Döngüsü Modellerini Geliştirme Modelleriyle Entegre Edilmesi	SP 2.1 Entegre yaşam döngüsü modelleri oluşturulması
	SP 2.2 Entegre yaşam döngüsü modellerini gözden geçirilmesi

Çizelge 7.11. Fonksiyonel olmayan test süreç alanı özel hedef ve uygulamaları

Özel Hedefler (SG)	**Özel Uygulamalar (SP)**
SG 1 İşlevsel Olmayan Ürün Riski Değerlendirmesi Yapılması	SP 1.1 İşlevsel olmayan ürün risklerini tanımlama
	SP 1.2 İşlevsel olmayan ürün risklerini analiz edilmesi
SG 2 Fonksiyonel Olmayan Bir Test Yaklaşımının Kurulması	SP 2.1 Test edilecek özellikleri tanımlama
	SP 2.2 Fonksiyonel olmayan test yaklaşımını tanımlama
	SP 2.3 Fonksiyonel olmayan çıkış kriterlerini tanımlama
SG 3 Fonksiyonel Olmayan Test Analizi ve Tasarımı Yapılması	SP 3.1 Fonksiyonel olmayan test koşullarını tanımlanması ve önceliklendirilmesi
	SP 3.2 Fonksiyonel olmayan test durumlarını tanımlamak ve öncelik vermek
	SP 3.3 Gerekli spesifik test verilerini tanımlanması
SG 4 Fonksiyonel Olmayan Test Uygulaması hazırlığı	SP 4.1 Fonksiyonel olmayan test prosedürlerini geliştirmek ve önceliklendirmek
	SP 4.2 Özel test verileri oluşturulması
SG 5 Fonksiyonel Olmayan Test Uygulaması Yürütülmesi	SP 5.1 Fonksiyonel olmayan test senaryoları yürütme
	SP 5.2 Fonksiyonel olmayan test olaylarını rapor edilmesi

Çizelge 7.12. Eş gözden geçirme süreç alanı özel hedef ve uygulamaları

Özel Hedefler (SG)	**Özel Uygulamalar (SP)**
SG 1 Gözden geçirme Yaklaşımının Kurulması	SP 1.1 İncelenecek iş ürünlerini tanımlanması
	SP 1.2 Gözden geçirme kriterlerini tanımlanması
SG 2 Gözden geçirme Gerçekleştirilmesi	SP 2.1 Gözden geçirme yürütülmesi
	SP 2.2 Test uzmanlarının temel test dokümanlarını gözden geçirmesi
	SP 2.3 Gözden geçirme verilerini analiz edilmesi

7.4. Değerlendirme Ölçüt Kriterleri

Mevcut test olgunluk seviyesini belirlemek ya da Seviye 2 ve Seviye 3 düzeyinde olan ya da olmak isteyen şirketlerin verilmiş olan yöntemi kullanırken dikkate alınması gereken kriterler aşağıda verilmiştir.

- Mevcut olgunluk seviyesi belirlemek için Seviye 2 de verilmiş tablo kullanılacaktır.
- Bu yöntem şirketteki çalışanlar tarafından kullanılabileceği gibi test deneyimi olan kişiler tarafından da kullanılabilir.
- Her soru için verilen cevapların ortalaması alınır.
- Ortalama olarak verilen cevapların toplamı alınarak maksimum puan olan 5 e bölünerek başarı yüzdesi belirlenir.
- Başarı yüzdesi %50'nin altında çıkan sonuçlar için Seviye 2 karşılanmamış yorumu yapılarak Seviye 1 düzeyinde olduğu sonucu çıkacaktır.

- Başarı yüzdesi %50'nin üstünde çıkan sonuçlar için Seviye 2 karşılandığının ve seviye 2 düzeyinde olduğu sonucu çıkacaktır.
- Seviye 2 düzeyinde olan şirketler Seviye 3 için verilen tabloyu kullanabileceklerdir.
- Seviye 3 için de ortalama olarak verilen cevapların toplamı alınarak maksimum puan olan 5 e bölünerek başarı yüzdesi belirlenir.
- Başarı yüzdesi %50'nin altında çıkan sonuçlar için Seviye 3 karşılanmamış yorumu yapılarak Seviye 2 düzeyinde olduğu sonucu çıkacaktır.
- Başarı yüzdesi %50'nin üstünde çıkan sonuçlar için seviye 3 düzeyi karşılanmış ve seviye 3 düzeyinde olduğu sonucu çıkacaktır.
- Bir sonraki seviye olan Seviye 4 için seviye yükseltme sürecine başlanabilir.

Tez kapsamında Seviye 4 ve Seviye 5 alınmadığı için bu seviyeler ile ilgili bilgi verilmemiştir.

7.5. Firma Bilgileri

Yöntemin uygulandığı şirketin ismi bilgi güvenliği gereği şirketin isteği üzerine gizli olacak şekilde verilmiştir.

Yazılım sektöründe telekomünikasyon alanında faaliyet gösteren bilişim, kablo yayıncılığı ve haberleşme alanlarında internet, telefon, TV hizmetleri vererek ülkemizde teknolojik değişik ve gelişime öncülük eden şirket 2005 yılında kurulmuştur. Şirket içerisinde farklı organizasyon şeması altında departmanlar bulunmaktadır. Her departman içerisinde yazılım geliştirme ve test süreçleri kendi bünyesinde olacak şekilde gerçekleştirilmektedir. Bu tez kapsamında bilişim tarafında yürütülmekte olan Cihaz Kiralama Stok Yönetimi (CKSY) projesi referans alınmıştır. Bir sonraki bölümde CKSY projesi ile ilgili bilgiler verilmiştir.

7.6. Cihaz Kiralama Stok Yönetimi (CKSY) Tanımı

Şirketin geliştirmiş olduğu web tabanlı sistem olan Cihaz Kiralama Stok Yönetimi (CKSY) projesi üzerinde uygulama gerçekleştirilmiştir. Bu bölümde CKSY sistemi ile ilgili bilgiler verilmiştir.

CKSY sisteminin temel kabiliyetleri cihaz kataloğunun yönetimi, depo yönetimi, bakım onarım yenileme sözleşme yönetimi ve yetki yönetiminden oluşmaktadır. CKSY sistemi içerisinde başlayan ve sonlanan süreçler olduğu gibi dış bir sistem tarafından tetiklenen ve CKSY sisteminde sonlanan süreçler de bulunmaktadır. CKSY sistemi bünyesinde yönetilen süreçler en üst seviyede süreçler üzerinden yönetilmektedir, bu süreçler entegre bir diğer sistemde yer alan iş emirleri ile kavramsal olarak benzeşmektedir. Hizmet satışlarının gerçekleştirildiği sistemde cihaz ile ilgili işlemlerde CKSY sistemi entegre olduğu için etkilenmektedir. Stok hareketleri cihazların özelliklerinin, durumunun, bulunduğu lokasyonun güncellenmesi gibi cihazları ilgilendiren işlemlerin gerçekleşmesini sağlar.

7.7. Problem Tanımı

Projeyi geliştiren organizasyonda benimsenen, sürdürülebilen ve etkili bir test süreci bulunmamaktadır. Yeni geliştirmeler, değişiklikler ya da hata düzeltmeleri süreç yönetiminde kullanılan uygulama ya da mail üzerinden paydaşlar tarafından ilerletilmektedir. Test süreci bu bağlamda kişilerin yetkinlikleri ve inisiyatiflerine bağlı şekilde ilerletilmektedir. Yani mevcut durumda bulgular tekrar edilemez ve herhangi bir kalite standardı veya süreci bulunmamaktadır. Uygulanan test aktivitelerinin temel amacı ise sadece uygulamaların çok büyük hatalar içermeden çalışmasıdır. Kaliteli ve etkili test yaklaşımının bütün çalışanlar tarafından benimsenmesi ve bu yaklaşımın yazılım geliştirme sürecine uygulanması proje planında gecikmeleri önlemesi, hata sayılarında azalması ve benimsenmiş etkin bir test sürecinin kurulması şirket için önemlidir. Canlıdan gelen hataları azaltılması, tekrarlı eforların önüne geçilmesi, yönetim baskısı giderilmesi ve müşteri memnuniyeti sağlanması ile oluşan maliyetlerden kurtulmak şirketin hedefleri arasındadır. Bu nedenle Test Olgunluk Entegrasyon modeli bu organizasyonda kullanılmak üzere seçilmiştir. TMMi modelinin süreç alanları Seviye 2 ve sonrasında Seviye 3 düzeyine çıkması için projenin test sürecine uygulanmıştır. TMMi modeli uygulanmadan önce CKSY sistemi için yapılan projenin test olgunluk seviyesi belirlenmiştir. CKSY sistemi için proje kapsamında belirtilmiş olan kaynaklar bir proje yöneticisi, bir iş analisti, bir test uzmanı, bir konfigürasyon yöneticisi ve üç geliştirici tanımlanmıştır.

7.8. Mevcut Test Olgunluk Seviyesinin Belirlenmesi

CKSY projesi üzerinde verilen mevcut olgunluk seviye belirlemesi için verilmiş olan yöntem uygulanmıştır. CKSY projesinde bulunan ekip tarafından verilen yanıtlar 5 üzerinden puanlanmıştır. Mevcut test olgunluk seviyesi belirleme için bu çalışma kapsamında önerilen seviye 2 değerlendirme ölçümü kullanılmıştır. Seviye 2 değerlendirme ölçümü için puanlamalar CKSY projesinde bulunan ekip tarafından yapılmıştır ve 5 üzerinden puanlanmıştır. Sorulara verilen puanların ortalama değerleri ve başarı puanı Çizelge 7.13' deki gibidir.

Çizelge 7.13. Seviye 2 değerlendirme ölçümü ile mevcut test olgunluk seviyesinin belirlenmesi

TMMi 2.Seviye	Verilen Puanların Ort.
1.Şirketinizde projeler ya da talepler için test politikası ve test stratejileri tanımlanıyor.	2,28
2.Test performans göstergeleri(test efor ve maliyeti, test süresi, bulunan hata sayısı, hata tespit yüzdesi, test kapsamı vb.) tanımlanıyor.	2,42
3.Test planlaması ve risk değerlendirmesi yapılıyor.	2,14
4.Test yaklaşımları (test edilecek öğeler, giriş ve çıkış kriterleri, durdurma ve başlatma kriterleri vb.) tanımlanıyor.	2,57
5.Plana dayalı test ilerleme süreci izleniyor ve raporlamalar yapılıyor.	2,71
6.Her test seviyesi(Birim, Entegrasyon, Sistem, Kabul testleri) açık bir şekilde tanımlanarak test hedefleri oluşturuluyor.	1,57
7.Test tasarım teknikleri(Kara kutu, Beyaz kutu, Deneyim temelli test teknikleri vb.) kullanılıyor.	1,28
8.Test durumları tanımlanıyor ve öncelik belirleniyor.	3,57
9.Hata yönetim süreci bulunuyor, test sırasında karşılaşılan hatalar raporlanıyor.	2,71
10.Yönetilebilir ve kontrol edilebilir test ortamı bulunuyor.	2,28
ORTALAMA	**2,353**
BAŞARI	**47,06**

Proje kapsamında çalışanlar tarafından sorulara verilen puanların ortalama değerleri ve Seviye 2 için başarı yüzdesi %47,06 olarak çıktığı görülmektedir. Bu sonuca göre CKSY projesi üzerinde test olgunluk seviyesi Seviye 1 olarak belirlenmiştir. Bundan sonraki bölümde Seviye 2 düzeyine geçiş ve seviye 2 değerlendirmesi için önerilen yöntemin uygulanışı ve sonrasında Seviye 3 düzeyine geçiş süreci verilecektir.

8. TMMi MODELİ İLE 2. ve 3. OLGUNLUK SEVİYELERİNE GEÇİŞ SÜRECİ ÜZERİNE BİR UYGULAMA

Olgunluk seviye belirleme yöntemi ile CKSY projesi mevcut olgunluk seviyesi belirlendikten sonra tez kapsamında olgunluk seviye 2 geçiş süreci ve sonrasında seviye 3 geçiş süreci bu bölümde anlatılacaktır. Olgunluk seviyesi olarak 2nci veya 3ncü düzeye çıkarmak isteyen firmalar için TMMi modeli seviye 2 ve seviye 3 konuları detaylı olarak verilmeye çalışılmıştır.

Telekomünikasyon alanında Türkiye çapında faaliyet gösteren ve bu sektörde gittikçe büyüyen ve abone sayısı artan şirketin geliştirmiş olduğu Cihaz Kiralama Stok Yönetimi (CKSY) projesinin test süreci olgunluk seviyesi önerilen model ile belirlenecek ve CKSY projesi üzerinde Test Olgunluk Model Entegrasyon (TMMi) modeli uygulanacaktır. TMMi modeli uygulamak isteyen şirketlerin bazıları resmi bir TMMi sertifikası almak isterken, bazısı da organizasyon süreçlerini test kalitesi açısından görmek ve iyileştirmek için isterler. Bu uygulamada organizasyonun test süreçlerinde ki kalite düzeyini ölçmek ve iyileştirmek benimsenmiştir. Resmi sertifikasyon süreci konusuna girilmemiştir. Önceki bölümde önerilen yöntem uygulanmış ve CKSY projesi ekibi tarafından verilen cevaplar sonucunda mevcut sistemin olgunluk seviyesi Seviye 1 olarak belirlenmişti. Sonraki bölümlerde olgunluk seviyesi 2 ve seviyesi 3 düzeylerine geçişler verilmiştir.

8.1. Seviye 1'den Seviye 2'ye Geçiş Süreci

8.1.1. Seviye 2 süreç alanları ve uygulamaları

CKSY sistemi üzerinde test olgunluk seviyesi belirlendikten sonra test olgunluk seviyesi 2 düzeyine çıkabilmek için TMMi modelinde verilmiş seviye 2 ye ait süreç alanları ve bu süreç alanlarına ait özel hedefler ve özel uygulamalar gerçekleştirilmiştir. Seviye 2 süreç alanları şunlardır:

- Test Politikası ve Stratejisi
- Test Planlaması

- Test İzleme ve Kontrol
- Test Tasarımı ve Uygulaması
- Test Ortamı

Süreç alanlarında özel hedefler ve özel uygulamalar bulunmaktadır. Bundan sonraki bölümlerde sırasıyla süreç alanlarına ait bilgiler verilecektir. Daha sonra ise bu süreç alanlarında bulunan özel hedefler (Specific Goal SG) ve bu özel hedeflere ulaşabilmek için yapılması gereken özel uygulamalar (Specific Practices SP) verilecektir. Özel hedefler SG ve numaralandırma ile takip edilecektir. Örneğin SG 1, SG 2 olarak isimlendirilecektir. Özel uygulamalar ise SP 1.1, SP 1.2 şeklinde isimlendirilecektir.

Test politikası ve stratejisi

Test politikası ve stratejisi süreç alanı, test politikasının ve stratejisinin uygulanması ve tanımını içerir. Test stratejisinde test seviyeleri belirlenir. Her test seviyesi için test hedefleri, sorumluluklar, temel görevler ve giriş / çıkış kriterleri tanımlanır. Test performansını ve test iyileştirme hedeflerinin başarısını ölçmek için, test performans göstergeleri tanımlanır ve uygulanır. Test politikası ve strateji süreç alanının amacı, test seviyelerinin kesin olarak tanımlandığı bir test politikası ve kurum çapında veya program çapında bir test stratejisi geliştirmek ve oluşturmaktır.

Test Politikası Süreç Alanı için Özel Hedefler(SG) ve Özel Uygulamalar(SP)
Test Politikası Belirleme Özel Hedefi (SG 1)

Bu süreç alanı aşağıdaki uygulamalar eşliğinde gerçekleştirilmiştir.

Test hedeflerini tanımlama (SP 1.1)

CKSY projesi kapsamında test amaçları belirlendi. İş analisti, geliştirici, müşteri (kullanıcı) ve proje yöneticisi ile geliştirilen ürün için gereksinim ve hedefler belirlenirken bilgi alınmıştır. Böylece paydaşlarında bu seviyedeki uygulama ile bilgilenmeleri sağlanmış ve iş gereksinim ve hedefler belirlenirken doğru bir yaklaşım sağlanmıştır.

Çizelge 8.1. CKSY sistemi için tanımlanan test hedefleri

Test Hedefleri	**Durum(Yapıldı/Yapılmadı)**
CKSY sistemini kullanıma uygun olarak doğrulamak.	Yapıldı
Operasyonda meydana gelebilecek hataları önlemek.	Yapıldı
Standartlara uygunluğu doğrulamak.	Yapıldı
Ürün kalitesi ile ilgili görünürlük sağlamak.	Yapıldı
Testler için gerekli test ortamını tanımlamak.	Yapıldı
Testler için gerekli verilerin hazırlanma şekli tanımlamak.	Yapıldı
Testler süresince uygulanacak teste giriş ve testten çıkış ölçütleri, testlerin geçme ve kalma ölçütleri tanımlamak.	Yapıldı
Test faaliyetlerinden sorumlu olacak personellere rol ve sorumlulukları hakkında bilgi vermek.	Yapıldı
Test yürütme süresini kısaltmak.	Yapıldı

Test politikasını tanımlama (SP 1.2)

Bu uygulama ile bir önceki uygulamada belirlenen test hedeflerine göre bu test hedefleri ile uyumlu olan bir test politikası tanımlanmıştır.
Test Politikası: "CKSY sistemi için bütün çalışanların ve organizasyonun benimseyeceği bir test süreci oluşturmak" olarak belirlenmiştir. Bu test politikası aşağıdaki bölümlerden oluşur:

- Testin tanımı
- *Hata ayıklama tanımı (hata tespiti ve çözümü*): Geliştirilmiş programlar aracılığı ile kod içerisinde hataları saptamanın, yazım hatalarını ya da mantık hatalarını çözmenin bunların sayılarını azaltmanın yoludur. Ürün geliştirilirken kod aşamasında geliştirici tarafında gözden geçirilerek ya da geliştirilen tool yardımıyla kod içerisinde hataları bulma işlemidir.
- Test ve test mesleği ile ilgili temel görüşler:
    - *Test:* Geliştirilen yazılımın işlevsel olarak çalıştığının ve bu çalışmayla kullanıcı gereksinimlerinin doğru bir şekilde sağlandığının gösterilmesidir.

- *Test mesleği:* Oldukça karmaşık yapı gibi görünen test organizasyonları gerçekte basit denilebilecek bir yapıya sahiptir. Test yönetimi altında incelenebilecek organizasyonda en temel görevler:

  - *Test Uzmanı (Tester)* : testi gerçekleştiren yürüten kişilerdir. Yazılım ürününün minimum sayıda hata içermesi ve müşteri isteklerine cevap verebilmesi ve gereksinimleri karşılayabilmesi için yazılım test aktiviteleri, yazılım geliştirme süreçlerinde en erken safhada başlamalıdır. Testçiler, yazılım geliştirme yaşama döngüsünün erken aşamalarında başlayan test işlemleri ile olası hataları yazılım geliştirme sürecinin daha başlarında bulmayı ve bulunan hataların düzeltilmesini amaçlar.

  - *Test lideri (Test leader):* Testi yapan kişilere /gruplara görevlerini dağıtan ve kendisi de test yapan kişilerdir.

  - *Test yöneticisi (manager):* test grubu/ gruplarını yöneten kişilerdir.

- *Hedefler ve testin katma değeri:* Yazılım geliştirme yaşam döngüsü, gereksinimlerin müşterilerden toplanarak analiz edilip daha sonra tasarım, geliştirme, test ve canlıya alma ve sonrasında bakım gibi aşamalardan oluşan bir döngüdür. Bu döngü içerisinde hata son aşamalarda bulunursa maliyeti daha yüksek olmaktadır. Son aşamalarda bulunan hatalardan doğacak maliyet, erken evrelerde bulunan hatanın maliyetine göre oldukça fazla olabilmektedir. Bu da testin erken aşamalarda başlamasının önemini arttırmaktadır.

- *Ulaşılacak kalite seviyeleri:* Mevcut seviye ile ilgili bilgiler önceki bölümlerde elde edilen sonuçlar referans alınarak paydaşlara aktarıldı. Bu kapsamda ulaşılacak kalite seviyeleri ile ilgili bilgilendirmeler yapılan toplantılar ile aktarıldı.

- *Test organizasyonunun bağımsızlık seviyesi:* Test organizasyonu yazılım yaşam döngüsünde olması gereken bir süreç ve bu varlığının olması için bağımlılık duymadığı konusunda bilgilendirmeler yapıldı. Testin bu seviyesi belirlenerek ve bilgilendirmeler yapılarak test ile ilgili olumsuz görüşlerin giderilmesi sağlanmıştır.

- *Üst düzey bir test süreci tanımı:* Üst düzey test tanımı kapsamında aşağıdaki durumlar ele alınarak açıklandı.
    - Neden test?
    - Test nedir?
    - 7 Test prensibi
    - Temel test süreçleri
    - Test psikolojisi

- *Test süreci iyileştirme için organizasyonel yaklaşım ve hedefler*

- Test edilecek ürün için projeye uygulanacak test tiplerinin belirlendiği anlayışa *test yaklaşımı* denir. Daha iyi bir test yaklaşımı ile
    - Yazılımda nelerin yapıldığı görülür ve yapılmaması beklenen özelliklerden ayırılır ve hata raporunda paylaşılır.
    - Yazılımdan beklenen fakat karşılayamadığı fonksiyonları ortaya çıkartılmış olur.

Geleneksel test yaklaşımının kullanıldığı süreçlerde karışık olmayan test caseleri hazırlanır ve sadece görsel hatalar bulunabilir. Sonuç olarak bulunması gereken hatalar ya da detaylandırılmış test caseleri yakalanacak hatalar bulunmamış olur ve gereksinimlerin tamamının karşılanıp karşılanmadığı saptanmamış olur. Bu durum ise yazılımın kalitesizliğine ve müşteri memnuniyetsizliğine neden olur.

Test politikasını ilgili kişilerle paylaşma (SP 1.3)

Bu uygulama kullanılarak belirlenen ve tanımlanan test politikası ve hedefleri proje kapsamında bulunan bütün paydaşlara sunularak açıklanmıştır. Proje ekibiyle toplantı yapılarak bu durumlar bildirilmiştir.

Test Stratejisi Belirleme (SG 2)

Test yaklaşımının ana hatlarını çizen ve projeye özgü hazırlanan kurallar bütünü olan bu süreç alanı aşağıdaki uygulamalar eşliğinde gerçekleştirilmiştir.

Genel bir ürün risk değerlendirmesi yapılması (SP 2.1)

Genel risk değerlendirmesine katkıda bulunacak paydaşlar: Analist ve talep yönetimi ekibi, yazılım geliştirme ekibi, test ekibi, konfigürasyon ve entegrasyon ekibi, destek ekibi, operasyon ekibi içerisinde birer kişi ile olacak şekilde belirlenmiştir.

Çizelge 8.2. CKSY sistemi için genel ürün riskleri

Genel Risk No	Genel Ürün Risk Tanımı	Olasılık(1-5)	Etki(1-5)	Risk faktörü(1-25)
R1	Sistemin sunucularının ayakta olmaması	1	5	5
R2	Ürünün doğru çalışamaması	1	4	4
R3	Sistemi kullananların sistem kullanımı hakkında yetkin olmamaları	4	3	12
R4	Güvenlik problemleri(rol ve yetki tanımlamalarına bağlı yapılacak işlemler)	2	3	6
R5	Firma imajını zedeleyecek görsel ve yazım hataları	1	1	1
R6	Gereksinimlerin tanımlı olmaması	2	2	4
R7	Yaşam döngüsünün tam olarak işletilememesi	2	3	6
R8	Yapılan planlamada yaşanacak aksaklıklar(Materyal, kaynak, ofis ortamı, network erişim sorunları vb.)	1	5	5
R9	Bakım onarım yenileme ve alım sözleşmelerinin gecikmesi	2	3	6
R10	Sözleşme içeriklerinde değişikliklerin yeni gereksinimler ortaya çıkarması	3	2	6
R11	Mevcut stok modülünde yer alan cihazların yeni sisteme aktarımında yaşanacak sorunlar(veri yapısı uyumsuzluğu, bozukluğu, eksikliği, vb.)	4	4	16
R12	Kaynakların doğru zamanda destek olamaması veya düşük sevide katılımı	3	4	12
R13	Veri aktarımı öncesinde stok sayımlarının doğru yapılamaması	4	4	16
R14	Yeni yapı ve iş süreçlerinin paydaşlar tarafından kullanılması esnasında yetersiz eğitim ve bilgilendirme nedeniyle yapılacak hatalar ve bu hataların düzeltilmesi	2	2	4

Genel ürün riskinin içeriği ve potansiyel sonuçları yukarıdaki gibi paydaşların katılımı ile netleştirilmiştir. Her bir genel ürün riskine ilişkin ilgili paydaşlar tanımlanmıştır:

- Ürün sahibi, proje yöneticisi, programcı vb. gibi sistem sorumluları
- Bakım, yardım masası vb. gibi ürün ile ilgili çalışmadan etkilenenler
- Son kullanıcılar, müşteriler vb. gibi sistemi kullananlar

Genel ürün riskleri yukarıdaki tanımlandıktan sonra analizi gerçekleştirilmiştir. Gerçekleşme olasılıkları ve gerçekleştiklerinde etkileri 1 ile 5 arasında derecelendirilerek risk faktörleri çıkarılmıştır. Böylece bu analiz referans alınarak sonraki uygulamalarda ürün riskleri tanımlanıp analizleri gerçekleştirilecek. Ürün risk analizi ile oluşturulacak test durumlarında odaklanılacak ve ağırlık verilecek durumlar netleştirilecektir.

Test stratejisi tanımlama (SP 2.2)

Test politikası temel alınarak bir test stratejisi belirlenmiştir. Test stratejisi, bir organizasyon için genel test gereksinimlerinden oluşur. Genel ürün risklerine hitap eden ve test politikasına uygun olan bu test stratejisi risklerin azaltılması için bir süreç hazırlar. Daha önceki uygulamalarda tanımlanan test politikası ve hedefleri incelenerek tanımlanan test politikası ve hedeflerine açık bir bağlantı sağlayan test stratejisi tanımlanmıştır. Test stratejisinde ele alınan durumlar:

- *Test seviyeleri (Birim, entegrasyon, sistem ve kabul testi):* Bu test seviyeleri için hedefler, sorumluluklar ve ana görevler tanımlanır. Test seviyeleri önceki bölümlerde detaylı verilmiştir.
    - *Birim testi:* Kod düzeyinde yazılım geliştiriciler tarafından gerçekleştirilir.
    - *Entegrasyon testi:* Entegre olan ve etkilenen sistemleri de kapsayacak şeklide test personeli tarafından gerçekleştirilir.
    - *Sistem testi*
    - *Kabul testi*
- Her bir test seviyesi için giriş ve çıkış kriterleri belirlenir.
- Teste başlama kriteri: Teste başlama kriterleri, geliştirilmekte olan ürünün test hazır olup olmadığını kontrol eden eksikleri belirlemek üzere hazırlanmış test kriterleridir. Ürünün teste hazır olup olmadığını anlamak için duman (smoke) testi yapılır. Bu kapsamda başlama kriterleri:
    - Personel, araç ve gereç ile materyal ihtiyaçları
    - Test edilecek ürünün hazır olması(Smoke Test)
    - Test datası

  - Testi sonlandırma kriterlerinin hazır olması

- Testi sonlandırma kriterleri: Testi tamamlanması ve ürün kullanıcıya sunulabilir durumda olması için gerekli kriterlere testi sonlandırma kriterleri denir. Bu kapsamda sonlandırma kriterleri:

  - Testlerin başarılı olması
  - Belirli bir kod kapsamının sağlanması
  - Hata dağılımının anlaşılması
  - Zamanın tamamlanması
  - Hata bulma sıklığının azalması
  - Kabul edilebilir riskler taşıması

- Testlerin gerçekleştirileceği ortamlar belirlenir. Geliştirme ortamı, test ortamı ve canlı ortamı olarak üç ortam bulunmaktadır. Geliştirme ortamı her bir geliştiricinin kendine özel çalıştıkları ortamlardır. Geliştirilen yer özelinde fonksiyonel test yapılır. Testten başarılı geçen bu geliştirilen kısımlar test ortamına deploy edilir. Burada bütün test seviyeleri için belirlenen test tipleri gerçekleştirilir. Test ortamı detaylı olarak sonraki bölümlerde ele alınacaktır.

- Her test seviyesi için otomasyon yaklaşımı yapılır. Oluşturulan test durumlarının manuel test edilebilirliği görüldükten sonra otomize edilebilecek caseler üzerinden analiz çalışması yapılır.

- Regresyon test yaklaşımı belirlenir ve sonraki seviyelerde ele alınacak test ortam kurulumu uygulamasından sonra gerçekleştirilir.

Test stratejisini ilgili kişilerle paylaşma (SP 2.3)

Bu uygulama kullanılarak belirlenen ve tanımlanan test stratejisi proje kapsamında paylaşılır.

Test Performans Göstergelerini Belirleme (SG 3)

Bu özel hedef için aşağıdaki uygulamalar gerçekleştirilmiştir.

Test performans göstergelerini tanımlama (SP 3.1)

Test süreci iyileştirme hedefleri incelenmiştir. Test politikası ve hedeflerinin gerektiği gibi olması adına geri bildirimlerde bulunulmuştur. Test performans göstergeleri:

- Test efor ve maliyeti
- Test süresi
- Bulunan hata sayısı
- Hata tespit yüzdesi
- Test kapsamı
- Test olgunluk seviyesi

Test performans göstergelerini uygulama (SP 3.2)

Tanımlanan performans göstergeleri uygulanmıştır.

- *Test efor ve maliyeti*: Test eforu gereksinimler ve kaynaklar referans alınarak a/g cinsinden efor belirlenmiştir.
- *Test süresi:* Proje planlamasında olan test süresi olarak belirtilmiştir.
- *Bulunan hata sayısı:* Test durumları gerçekleştirilirken karşılaşılan hatalar, geliştiricilere iletildi. Çözümler sağlandıktan sonra tekrarlı test ve sonrasında

sistemin genel gereksinimleri için oluşturulan regresyon test durumları gerçekleştirildi. Hata yönetimi JIRA toolu ile gerçekleştirilmiştir.

- *Hata tespit yüzdesi:* (Bulunan Hata Sayısı/Test Durum sayısı) formülüne göre değerlendirilmiştir.

- *Test kapsamı:* Her test seviyesi için kapsamlar belirtildi. Birim Testi için test durumları oluşturuldu. Bu test durumları gerçekleştirildikten sonra entegrasyon testleri, sistem testleri ve kabul testleri gerçekleştirilmiştir.

- *Test olgunluk seviyesi:* Test olgunluk seviyesi mevcut sürecin analizi yapıldıktan sonra önerilen model sonrasında Seviye 1 olarak belirlenmişti. Seviye 2 ve sonrasında Seviye 3 düzeyine çıkmak hedef olarak belirlenmiştir.

Test Planlaması

Test planlaması, test objesi üzerinde bir ürün risk değerlendirmesi gerçekleştirilmesini ve tanımlanan risklere göre farklı bir test yaklaşımının tanımlanmasını içerir. Ayrıca, yapılacak testler için tahminlerin geliştirilmesini, gerekli taahhütlerin oluşturulmasını ve testin yönlendirilmesi ve yönetilmesi için planın tanımlanmasını ve sürdürülmesini içerir. Her bir test seviyesi için bir test planı gereklidir. TMMi 2 seviyesinde test planları tipik olarak test seviyesi başına geliştirilir. Test planlamasının amacı, tanımlanan risklere ve tanımlanmış test stratejisine dayanan bir test yaklaşımı tanımlamak ve test faaliyetlerini yürütmek ve yönetmek için iyi kurulmuş planlar oluşturmak ve sürdürmektir.

Test Planlaması Süreç Alanı için Özel Hedefler(SG) ve Özel Uygulamalar(SP)

Ürün Risk Değerlendirmesi (SG 1)

Ürün risk değerlendirmesi CKSY sistem gereksinimleri dikkate alınarak aşağıdaki uygulamalar ile oluşturulmuştur.

Ürün risk kategorileri ve parametreleri (SP 1.1)

Ürün risk kategorileri genel ürün risk değerlendirmelerinden yola çıkılarak tanımlanmıştır.

Çizelge 8.3. Ürün risk kategorileri ve risk tanımları

Fonksiyonel riskler	Mimari riskler	Değişiklik riskleri
Sisteme giriş sorunları	Entegre olduğu sistem ile yaşanacak entegrasyon sorunları	Deploy sonrası sistemin mevcut yapısında yaşanacak sorunlar
Toplu işlemlerde yavaşlık ve kesinti problemleri	Hizmet satış sonrası cihaz durumunun değişmeme riski	Talep ve gereksinimlerin değişmesi, yeni isteklerin gelmesi
Yanlış cihaz değişimi yapılması	Hizmet iptali sonrası cihaz durumunun değişmeme riski	
Cihaz katalog yönetimi ekranlarında butonların aktif ya da pasif olduğu durumlar	Cihaz kiralama siparişlerinde cihaz kiralama sürecinin CKSY' de işlememe riski	
Cihaz tanımlama riski	Cihaz değişim iş emirlerinde CKSY ya da entegre olduğu sistemde yaşanacak sorun	
Cihaz özellik güncelleme riski	Cihaz iade durumlarının işletilememe riski	
Sevk süreci riski	Cihaz arıza sürecinde yaşanacak sorun	
Menüler arası geçişlerde yavaşlık	Depo yönetiminde kullanıcı işlemleri için yetki ve rol tanımlamalar riski	
Sayfa yüklemelerin gecikmesi	Cihaz yönetiminde kullanıcı işlemleri için yetki ve rol tanımlamalar riski	
	Ekranda görünür alanlardan alan ve uyarı mesaj içeriklerinde yazım hataları	

Kullanılan etki faktörleri;

- Kritik olan alanlar
- Görünür olan alanlar
- En çok kullanılan alanlar
- Yapılan iş açısından önemi
- Tekrarlı olacak eforlarda çalışma maliyeti

Kullanılan olasılık faktörleri;

- Karmaşıklık
- Müşteri isteklerine göre değişiklik yapma sıklığı
- Yeni oluşan teknolojiler ve oluşan metotlar
- Oluşan zaman baskısı
- Ara yüz

- Boyut farklılığı
- Tespit edilen hata geçmişi
- Gereksinimlerin kalitesi

Ürün riskleri (SP 1.2)

Her bir risk için paydaşlar tanımlanır. Risk değerlendirme içerisinde bulunan paydaşlar belirlenmiştir. Paydaşlar, sistemi kullananlar (son kullanıcılar), sistemden sorumlu olanlar (ürün sahibi, proje yöneticisi, programcı), ürün doğru çalışmadığında etkilenenler (bakım, yardım masası) arasından seçilmiştir. Ürün riskleri belirlenirken beyin fırtınası tekniği kullanılmıştır. Risklerin gerçekleşme durumunda karşılaşabilecek durumlar ve riskler için belirtilen olasılıkların gerçekleşmesi durumunda verilecek etkiler belirtilmiştir. Ör. Sisteme giriş sorunu yaşanma olasılığı çok düşük iken etkisi yüksek olacaktır. Bir kişi yönetim seviyesinden olmak üzere toplamda yedi paydaş belirlenmiştir. Bir proje yöneticisi, bir iş analisti, bir yazılım mimarı, üç programcı ve bir de test uzmanı seçilmiştir. Tespit edilen ürün riskleri ile testlerin önceliklendirilmesi, test edilebilirliği ve oluşturulacak test durumları belirlenir.

Ürün riskleri analizi (SP 1.3)

Belirlenen ürün risklerini, önceden tanımlanmış parametreleri kullanarak, Olasılık ve etki ile analiz edilmiştir. Gerçekleşme olasılığı düşük ama etkisi büyük olacak olan riskler için oluşturulacak test caseler önceliklendirilir. Bir gereksinim / ürün riskleri izlenebilirlik matrisi oluşturulur. Ürün risklerinin eksiksizliği, kategorisi ve öncelik seviyesi hakkında paydaşlarla görüşme ve anlaşma sağlamak adına belirli periyotlarla toplantılar yapılır. Değişen gereksinimlerde ürün riskleri için gözden geçirmeler paydaşlar ile yapılmıştır.

Çizelge 8.4. CKSY sistemi için ürün risk analizi

Risk No	Risk tanımı	Çıkma Olasılığı(1-5)	Etkisi(1-5)	Risk Değeri(1-25)
R1	Sisteme giriş sorunları	1	5	5
R2	Toplu işlemlerde yavaşlık ve kesinti problemleri	2	1	2
R3	Yanlış cihaz değişimi yapılması	2	2	4
R4	Cihaz katalog yönetiminde veri girişinde format hatalarının olması	1	3	3
R5	Cihazların stoka yanlış girişleri	2	2	4
R6	Cihaz özelliklerini güncelleyememe	1	1	1
R7	Cihazların yanlış depolara sevkleri ya da yanlış cihazların depolara sevkleri	3	1	3
R8	Menüler arası geçişlerde yavaşlık	2	1	2
R9	Sayfa yüklemelerin gecikmesi	2	1	2
R10	Entegre olduğu sistem ile yaşanacak entegrasyon sorunları	4	5	20
R11	Hizmet satış sonrası cihaz durumunun değişmeme riski	4	5	20
R12	Hizmet iptali sonrası cihaz durumunun değişmeme riski	4	5	20
R13	Cihaz kiralama siparişlerinde cihaz kiralama sürecinin CKSY de işlememe riski	4	5	20
R14	Cihaz değişim iş emirlerinde CKSY ya da entegre olduğu sistemde yaşanacak sorun	4	5	20
R15	Cihaz iade durumlarının işletilememe riski	1	3	3
R16	Cihaz arıza sürecinde yaşanacak sorunlar	2	4	8
R17	Depo yönetiminde kullanıcı işlemleri için yetki ve rol tanımlamalar riski	1	2	2
R18	Cihaz yönetiminde kullanıcı işlemleri için yetki ve rol tanımlamalar riski	1	3	3
R19	Ekranda görünür alanlardan etiket ve uyarı mesaj içeriklerinde yazım hataları	1	1	1
R20	Deploy sonrası sistemin mevcut yapısında yaşanacak sorunlar	1	5	5
R21	Talep ve gereksinimlerin değişmesi, yeni isteklerin gelmesi	4	5	20

Bir Test Yaklaşımının Kurulması (SG 2)

Belirlenen ürün risklerine dayanan test yaklaşımı aşağıdaki uygulamalar ile sağlanarak bu süreç alanı sağlanmıştır. Bütün bileşenleri içerisine katarak oluşturulan projede daha başarılı test sonuçları yakalamak için kullanılan test tiplerinin gruplandırıldığı test anlayışıdır.

- *Analitik Yaklaşım:* Web tabanlı geliştirilen sistemlerde genel olarak internet üzerinden yapıldığı için bazı riskler vardır. Örneğin; güvenlik problemleri, firma imajını zedeleyecek görsel ve yazım hataları.

- *Dinamik Yaklaşım:* Gereksinimlerin olmadığı, planlamanın tam olarak yapılmadığı hızlı bir yazılım geliştirme ortamının bulunduğu durumlarda kullanılan yaklaşım türüdür.

Analitik ve dinamik test stratejilerinden oluşan; *Risk Temelli - Dinamik* test yaklaşımı belirlenmiştir.

- Hata verme olasılığı bulunan alanlara ve daha kritik alanlara daha fazla test yapmak.
- Keşif temelli testler ile hata tahminlerinde bulunarak, ilgili alanlara yoğunlaşmak.
- Gereksinimlerin doğrulanmasını sağlayan test durumları oluşturmak.

Test edilecek öğeleri ve özellikleri tanımlama (SP 2.1)

Ürün risklerini göre test edilen ya da test edilemeyen kalemlerine ayrılarak test edilecek ürün riskleri belgelenmiştir.

Çizelge 8.5. CKSY sistemi test edilecek öğeler

Cihaz Kiralama ve Stok Yönetimi Modülü Geliştirmeleri	Kullanıcı Tarafından Tetiklenenler(Servis Seviyesinde & Ekran Barındıran)	Süreç Tarafından Tetiklenenler(Kaynak Seviyesinde & Arka Planda Oluşan)
Depo Yönetimi	Cihaz tanımlama(Dosya Yükleme&Validasyon)	Cihaz Satış
Rol ve Yetki Yönetimi	Sevk	Cihaz Kiralama
Cihaz Yönetimi(katalog)	Bakım Onarım Yenileme(BOY)	Cihaz Değişim
Bakım Yönetimi	Garanti	Arıza
Hak ediş Yönetimi	Cihaz Durum Güncelleme	İptal
Raporlama	Cihaz Özellik Güncelleme	Hizmet İade
Merkezi Düzeltme		Taahhütlü Şebeke Dışı Şartlı İptal
İş emri Yönetimi		Devir
		Nakil

Test yaklaşımını tanımlama (SP 2.2)

Bu uygulama ile test tasarım tekniği seçilir. Sistem tipi, risk seviyesi, risk türü, test edenlerin bilgisi, deneyim ile hataların bulunması vb. kriterleri kullanılarak test tasarım tekniği seçilmiştir. Statik test teknikleriyle kod ve doküman incelemesi yapılmıştır. Dinamik test teknikleriyle ise kod çalıştırıldı ve kara kutu test teknikleri çalışan koda uygulanmıştır.

Regresyon testi için yaklaşımı tanımlanarak gerçekleştirilecek test durumları, manuel ve otomasyon edilecek test durumlarının belirlenmiştir. Test araçları olarak web tabanlı "TestRail" toolu kullanılmıştır.

Giriş kriterlerini tanımlama (SP 2.3)

Test süreci ile ilgili giriş kriterler:

- Bir önceki test seviyesinden bir test özeti raporunun kullanılabilirliği
- Gereksinimlere göre bir test ortamının kullanılabilirliği
- Belgelerin kullanılabilirliği, Test sürüm notları, kullanım kılavuzu

Ürün kalitesine ilişkin giriş kriterleri aşağıdaki gibi:

- Başarılı bir giriş testi (Smoke Testi)
- Olağanüstü hataların olmaması
- Tüm olağanüstü hatalar analiz edilmesi ve çözülmesi

Giriş kriterleri, özellikle giriş kriterlerini karşılamaktan sorumlu olan paydaşlarla birlikte gözden geçirilmiştir.

Çıkış kriterlerini tanımlama (SP 2.4)

Test işlemiyle ilgili çıkış kriterleri:

- Yürütülen testlerin yüzdesi (başarıyla)
- Her bir test öğesi için kapsama yüzdesi, Kod kapsamı veya gereksinim kapsamı
- Onaylanmış bir test özeti raporunun kullanılabilirliği

Ürün kalitesine ilişkin çıkış kriterleri:

- Tüm yüksek öncelikli ürün risklerinin azaltılması

- Hata tespit oranının bir eşiğin altına düşmesi
- Olağanüstü hataların sayısı (öncelik seviyesine göre)

Durdurma ve yeniden başlatma kriterlerini tanımlama (SP 2.5)

Test görevlerinin ve özelliklerinin test görevlerinin tümünü veya bir kısmını durdurmak için kullanılan durdurma kriterleri:

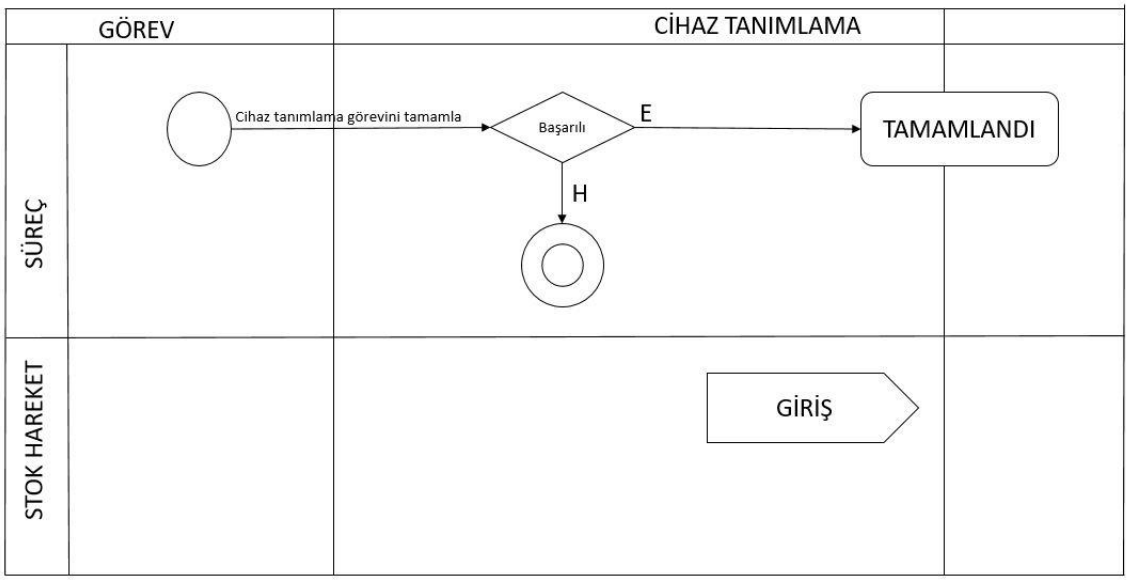
- Kritik hataların sayısı
- Tekrar üretilemeyen hataların sayısı
- Test ortamları nedeniyle test yürütme ile ilgili sorunlar

Test Tahminlerini Oluşturma (SG 3)

Bu süreç aşağıdaki uygulamalarla sağlanmıştır.

Üst düzey bir iş analizi yapısı kurma (SP 3.1)

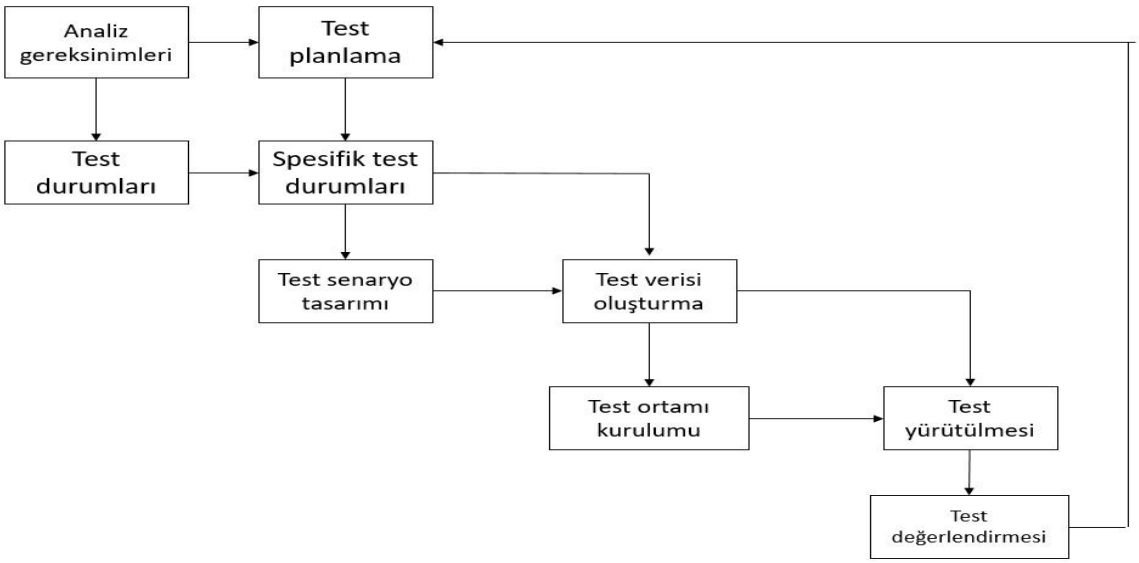
Üst düzey bir iş analizi yapısı iş analisti tarafından gerçekleştirilir. İhtiyaçların yani gelen isteklerin gereksinim ve çözüm analizleri iş analisti tarafından iş analiz kriterlerine göre yapılmaktadır. Bunun yanında gereksinimlerin test edilebilir olgunlukta olması için test uzmanı ile birlikte toplantılar yapılarak, gereksinimler gözden geçirilerek (statik test teknikleri) sağlanmıştır. İş analizi için süreç akış şemaları, iş akış diyagramları oluşturulmuştur. Örnek olarak Şekil 8.1’ de cihaz tanımlama süreci akış şeması verilmiştir.



Şekil 8.1. Cihaz tanımlama süreci

Test döngüsü tanımlama (SP 3.2)

Test planlama, test hazırlığı ve test yürütme aşamaları test yaşam döngüsüne tanımlanmıştır. Bir önceki uygulama ile kurulan üst düzey iş analizi yapısı yaşam döngüsüne dâhil edilmiştir.



Şekil 8.2. Test yaşam döngüsü

Test efor ve maliyet için tahminleri belirleme (SP 3.3)

Test durum sayılarının, test verilerinin yoğunluğu ve gereksinim sayıları için tahmin yaklaşımı uygulanmıştır. Test öğelerinin karmaşıklığı, yeniden kullanılma seviyeleri ve belirlenen ürün risklerinin öncelikleri paydaşlarla birlikte toplantılar yapılarak tahminler belirlenmiştir. Tahminler yapılırken geçmiş veriler ve daha önce onaylanmış modeller referans alınmıştır.

Tahminleri etkileyen faktörler:

- Test araçlarının kullanımı(Kara kutu test tasarım teknikleri için kullanılması planlanan araçlar ve birim testi için kullanılması planlanan araçlar)
- Daha önceki test seviyelerinin kalitesi
- Test temeli kalitesi
- Geliştirme ortamı
- Test ortamı
- Önceki projelerin yeniden kullanılabilir test yazılımlarının kullanılabilirliği
- Test edicilerin bilgi ve beceri seviyesi

Test tahmini için "Geliştirme eforuna karşı test eforu oranı" modeli kullanılmıştır. Test çabası ve maliyeti tahmin edilirken aşağıdaki destekleyici altyapılar:

- Test ortamı
- Kritik bilgisayar kaynakları
- Ofis ortamı
- Test araçları

Test sürecinin planlaması açısından testlere başlamadan önce tahminler yapılması önemlidir. Kaliteli bir test süreci için gereken minimum süre ve test organizasyonunda bulanacak çalışan sayısı tahmin sürecindeki önemli tahminlerdir. Test süresinin tahmin edilmesi için göz önünde tutulması gereken parametreler; test durum sayısı ve gereksinim sayılarıdır. Buradan yol çıkarak tahmini test süresi:

test süresi = (test_case_sayısı X test_işletim_süresi) / testci_sayısı

Yazılımcı başına düşen testçi sayısı tahmin edilmesi önemli olan başka bir konudur. Yazılımcı başına düşen testçi sayısı, yazılımcının olgunluk seviyesi, projenin büyüklüğü, zorluk derecesi, riskler, güvenlik kriterleri, kullanıcı profili gibi geniş kriterlerden etkilenir.

- Testçi Sayısı (proje konusu, Proje riski, Ürün riski)
- Test süresi (Test case sayısı, Gereksinim sayısı, İşletim süresi)
- Karmaşıklığı/Zorluğu (Kullanılan methotlar, yeni teknolojiler, güvenlik araçları)

Test Planı Geliştirilmesi (SG 4)

Test planı aşağıdaki uygulamalar ile yapılmıştır. Test plan dokümanı şablonu Ekler bölümünde verilmiştir.

Test programını belirleme (SP 4.1)

Kaynaklar, gerekli girdiler ve süre kısıtlamaları, test için bağımlılıklar tanımlanmıştır.
Test programında aşağıdaki maddeler bulunur:

- Geliştirme süresi
- Geliştirme yapan kişilerin yetkinlikleri
- Geliştirmenin doğru yapılıp yapılmamasını kontrol eden smoke testlerinin başarılı olması
- Test ortamı erişimi
- Canlı ortam ile uyumlu olması vb. gibi değişkenler

Test personelini planlama (SP 4.2)

İş analiz yapısına, test tahminine ve test programına göre personel gereksinimleri tanımlanmıştır. Test görevlerinin yerine getirilmesi için gerekli bilgi ve beceriler belirlenmiştir.

Test görevi için çalışacak kişilerin özellikleri, gereksinimleri ve sorumlulukları:

- Test mühendisi, geliştirilen yazılım ürününde çeşitli testler yaparak müşteri isteklerinin karşılandığının kontrolünü yapan kişilerdir.
- Test ortamı ile ideal test ortamı arasındaki farkı bilen kişilerdir.
- Bu farklılıktan dolayı ortaya çıkabilecek sorunları analiz edebilen kişilerdir.

Testçinin ihtiyaçları:

- Sistem ya da sistemler ile ilgili süreçlere ait genel bilgiler
- Yazılım ürünü hakkında genel bilgi
- Testlerde bulunan hataları ve test kapanış raporlama yetkisi
- Teste gelmeden önce kodlamanın bitmiş olması
- Hataları bulma hakkı
- Sistemde çıkabilecek hatalar ile ilgili öngörüde bulunma
- Test süreçlerini iyileştirme yetkisi
- Uzmanlık alanı olarak kabul görmesi
- Test planı oluşturulması ve detaylandırılması ile ilgili söz hakkı

Testin yapılabilmesi için birtakım hakların ve yetkilerin şirket organizasyonunda bulunması gerekmektedir. Bu yetkiler ile testçiler, daha iyi test yapabilir, bulunan hataları objektif bir şekilde raporlama yapabilir ve hataların çözülmesi sürecinde aktif olabilir. Test süreçlerinde saptanan hataları düzeltme yetkisiyle birlikte daha iyi test yapabilmenin önü açılacaktır.

Testçinin sorumlulukları:

- Hataların tarafsız ve gerçekçi bir şekilde raporlanması
- Yazılımı test ettiğinin bilincinde olması
- Riskleri tarafsız değerlendirmesi

- Test planını takip etmesi
- Yürüttüğü testlerde karşılaştığı durumlar ile ilgili gerçekleri paylaşması
- Hatayı raporlamadan önce kontrol etmesi
- Hataları önceliklendirmesi
- Yazılımcıyı test etmemesi gerektiğini bilmesi

Paydaş katılımını planlama (SP 4.3)

Paydaş katılımı her uygulama sonrasında kısa toplantılar olacak şekilde planlanır.

Test projesi risklerini tanımlama (SP 4.4)

Proje riskleri aşağıdaki teknikler kullanılarak belirlenir:

- Beyin fırtınası
- Uzman görüşmeleri
- Kontrol listeleri

Bu tekniklerden faydalanarak riskler aşağıdaki gibi oluşturulur.

- Süreçteki değişiklikler
- Test ortamında çalışmayan test araçları
- Test durumlarının yeniden yazılmasına sebep olabilecek değişiklikler
- Test ortamındaki hatalar
- Kaynak eksilmeleri: testçi, test analisti vb.
- Donanım eksiklikleri
- Teknik problemler: tasarım, kod vb.

Test planının oluşturulması (SP 4.5)

Daha önceki uygulamalarda belirlenen adımların tamamı test planını oluşturmaktadır. Test planı elemanları:

- Test planı tanımlayıcısı
- Genel bir giriş
- Test stratejisine ve gerekçesine uygunsuzluklar
- Test edilecek ürünler (öncelik seviyesi dâhil) ve test edilmeyecekler
- Test edilecek özellikler (öncelik seviyesi dâhil) ve test edilmeyecekler
- Test yaklaşımı (ör. Test tasarım teknikleri)
- Giriş ve çıkış kriterleri
- Durdurma ve yeniden başlatma kriterleri
- Yaşam döngüsü ve görevleri
- Çevresel ihtiyaçlar ve gereksinimler (ofis ortamı dâhil)
- Personel ve eğitim ihtiyaçları
- Paydaş katılımı
- Test tahmini
- Test programı
- Test projesi riskleri ve ihtimalleri

Test İzleme ve Kontrol

Süreç izleme ve kontrol süreci, test ilerlemesini ve ürün kalitesini belgelendirilmiş tahminlere, taahhütlere, yapılan planlamalara ve beklentilere karşı izlemeyi, test ilerlemesini ve ürün kalitesini paydaşlara bildirmeyi, kontrol önlemlerini almayı (gerektiğinde düzeltici eylemleri) ve kapatmak için çözen aktivteleri yönetmeyi içerir. Test izleme ve kontrol süreç alanının amacı, test ilerlemesinin ve ürün kalitesinin anlaşılmasını sağlamaktır.

Test İzleme ve Kontrol Süreç Alanı için Özel Hedefler(SG) ve Özel Uygulamalar(SP)

Plana Dayalı Test İlerleme Sürecini İzleme (SG 1)

Testin gerçek gelişimi ve performansı izlenerek test planındaki değerlerle karşılaştırılır. Aşağıdaki uygulamalar uygulanmıştır.

Test planlama parametrelerini izleme (SP 1.1)

Test planlama da tanıtılan maddelerin uygulama ile birlikte izlenir. Test plan parametreleri test planlama bölümünde bulunmaktadır.

Sağlanan ve kullanılan test ortamı kaynaklarını izleme (SP 1.2)

Bu uygulama ile test ortamlarında deployлardan sonra kısa sürelide olsa erişim sorunları ve ana test durumlarında hatalar tespit edildiği gözlenir. Test ortam kaynakları detaylı olarak sonraki bölümlerde verilecektir.

Test projesi risklerini izleme (SP 1.3)

Daha önce tanımlanmış olan proje riskleri izleme sonrası gözlemlenen durumlar aşağıdaki gibidir:

- Süreçteki değişiklikler
- Test ortamında çalışmayan test araçları
- Test caselerin yeniden yazılmasına sebep olabilecek değişiklikler
- Test ortamındaki hatalar
- Kaynak eksilmeleri
- Donanım eksiklikleri
- Teknik problemler

Test ilerlemesi değerlendirmelerini yönetilmesi (SP 1.4)

Test ilerleme raporları, durum raporları paylaşılır. Aşağıdaki bölümlerde testlerde bulunan hatalar ve test durumlarının raporlamasında kullanılabilecek tablolar verilmiştir.

Plan ve Beklentilere göre Ürün Kalitesini İzleme (SG 2)

Aşağıdaki uygulamalar ile gerçekleştirilmiştir.

Giriş kriterlerine göre kontrol (SP 2.1)

Test planında belirlenen giriş kriterlerine karşı durum kontrol edilir.

İzleme Hataları (SP 2.2)

Beklentilere karşı test sırasında tespit edilen hatalar şirkette kullanılan JIRA toolunda açılmaktadır ve takibi bu tool ile yapılabilmektedir. Ayrıca test kapanışlarında ve günlük test raporları şablonları kullanılarak test durumları takibi yapılır. Günlük paylaşılan test durum tabloları olarak aşağıda oluşturulmuş tablolar kullanıldı.

Çizelge 8.6. Örnek defect çözüm bilgileri tablosu

Proje Adı	Önem Derecesi	Toplam hata sayısı	Hata Çözüm Süresi /g	Hata çözülmesi beklenen süre bilgisi /g
CKSY	High	12	5	2

Çizelge 8.7. Açık hata (defect) tablosu

Proje Adı	Hata No	Hata Durumu	Hata Önem Derecesi	Atanan	Belirleme Tarihi	Hata Açıklama(Adı)	Hata Tipi	Sistem Bilgisi
CKSY	1	Açık	high	Gökhan Ş.			Code defect	CKSY
CKSY	2	İşlemde	high	Gökhan Ş.			Req. defect	CKSY

Çizelge 8.8. Hata (Defect) durumları

Önem Derecesi	**Closed**	**Deferred**	**Deferred&Closed**	**Fixed**	**Open**	**Rejected**	**Rejected&Closed**	**Toplam**
Orta (Medium)								
Yüksek (High)	7	0	0	5	2	1	2	**17**
Düşük (Low)								
Kritik (Critical)								

Çizelge 8.9. Test durumları

Failed	**Blocked**	**N/A**	**No Run**	**Not Completed**	**Passed**	**<Total>**
1	1	0	5	0	15	22

Çizelge 8.10. Planlanan ve gerçekleşen durum tablosu

Günlük Hedef	Tarih	Planlanan	Gerçekleşen (Passed+Failed)	Passed
5		5	6	4
5		4	4	4
6		6	6	5
6		6	6	6

Çizelge 8.10 kullanılarak tablodaki verilerin grafikleri ile de takip edilebilir bir raporlama da kullanılabilir.

Çizelge 8.11. Zaman kaybı

Açıklama (Solution)	Bloke olunan Süre (Gün)	Projeye Etkisi (Gün)
Problem 1		
Problem 2		
Toplam		
Açıklama (Ortam sorunları)	Bloke olunan Süre (Gün)	Projeye Etkisi (Gün)
Problem 1		
Problem 2		
Toplam		

Ürün risklerini izleme (SP 2.3)

Ürün riskleri, test planında belirtilenlere karşı izlenir. Daha önce belirlenen ürün riskleri gerçekleştirilen testlere göre kontrol edilir ve paylaşılır.

Çıkış kriterlerini izleme (SP 2.4)

Test planında belirtilenlere karşı çıkış kriterlerinin durumu izlenir.

Durdurma ve tekrar başlama kriterlerini izleme (SP 2.5)

Durdurma kritelerlerini karşılayan durumların olduğu ve testlerin durdurulduğu gözlemlenir. Test ortamı ve testlerde bulunan hataların çözülmesi aşamasında testler durdurulur ya da hatanın etkilemediği test durumları varsa onlara devam edilir.

Ürün kalitesi gözden geçirmeleri (SP 2.6)

Ürün kalitesi incelemeleri, paydaşların bilgilendirilmesi için test ekibi üyeleriyle ve harici olarak test dışındaki paydaşlarla yapılır. Bu değerlendirmeler düzenli olarak yapılan gayri resmi değerlendirmelerdir.

Kapanışı Düzeltici Faaliyetlerin Yönetilmesi (SG 3)

Test ilerlemesi izleme ve kontrol için kapanış faaliyetleri aşağıdaki uygulamalar gerçekleştirilmiştir.

Analiz Sorunları (SP 3.1)

Geliştirme ve test aşamalarında yeni gelen gereksinimler ve güncellenen gereksinimler gibi sorunlar gözlemlenebilir. Analiz sorunları olarak gereksinim gözden geçirmelerinde gereksinimler ile ilgili tespit edilen hatalar ilgili paydaşlara iletilir ve çözümleri sağlanarak tekrar gözden geçirilmesi sağlanır. Analiz sorunları tespit etme yöntemleri Eş gözden geçirme süreç alanında detaylı ele alınmıştır.

Düzeltici Eylem gerçekleştirilmesi (SP 3.2)

Düzeltici eylemler geliştirici ve analistler tarafından gerçekleştirilecek şekilde planlanır ve tespit edilen sorunlar ilgili paydaşlara iletilerek çözümleri sağlanabilir.

Düzeltici Eylem Yönetimi (SP 3.3)

Düzeltici eylemlerin yönetimi ve kapanış yönetiminde gerçekleşen çözümler ile değişen, güncellenen, eklenen ya da silinen test durumları test uzmanı tarafından gerçekleştirilerek yönetimi sağlanır.

Test Tasarımı ve Yürütülmesi

Test tasarımı ve uygulaması süreç alanı, test koşulları ve test durumlarını türetmek ve seçmek için test tasarım tekniklerinin uygulanmasını içeren test hazırlama aşamasını ele almaktadır. Ayrıca belirli test verilerinin oluşturulmasını, belgelenmiş test prosedürlerini ve olay yönetimini kullanarak testlerin yürütülmesini de ele alır. Test tasarım ve uygulamasının amacı, test tasarım özellikleri belirleyerek, test tasarım teknikleri kullanarak, yapılandırılmış bir test yürütme süreci gerçekleştirmesi ve test durumlarını kapatmayı yönetilmesi yoluyla, test tasarımı ve yürütme süresince test süreci yeteneğini geliştirmektir.

Test Tasarımı ve Yürütülmesi Süreç Alanı için Özel Hedefler(SG) ve Özel Uygulamalar(SP)

Test Tasarım Teknikleri Kullanarak Test Analizi ve Tasarımı Yapılması (SG 1)

Daha önceki süreçlerde tanımlanan test tasarım teknikleri kullanılarak bu süreçte test durumları aşağıdaki uygulamalar ile tasarlanır ve oluşturulur.

Test koşullarının tanımlanması ve öncelik verilmesi (SP 1.1)

Test koşulları aşağıdakiler temel alınarak incelenmiş ve analiz edilmiştir.

- Gereksinimler
- Mimari
- Tasarım
- Ara yüz özellikleri

Kara kutu testi tasarım tekniklerinden eşit bölümlere ayırma, sınır değer analizi ve karar tabloları kullanılır. Beyaz kutu testi tasarım tekniklerinden bildirim testi, karar testi ve durum testi kullanılır. Bu test teknikleri yazılım test süreçleri bölümünde detaylı olarak ele alınmıştır. Test koşulları test tasarım teknikleri kullanılarak elde edilir. Test koşulları belirlenen ürün risklerine göre önceliklendirilir.

Test durumlarını tanımlanması ve öncelik belirlenmesi (SP 1.2)

Test tasarım teknikleri kullanılarak test senaryoları oluşturulmuştur. Test durum dokümanında yer alan test durum isimleri tablosu aşağıdaki gibidir.

Çizelge 8.12. Test durum dokümanında yer alan test durumları

Gereksinim No	Kullanım Durumu ID	Kullanım Durumu Adı	Test Durumu ID	Test Durum Adı	Ön Koşul	Öncelik
Gereksinim_1			TD.01	Özellik güncelleme yetki tanımı yapılması		
Gereksinim_1			TD.02	Özellik güncelleme tüm depolar yetki tanımı yapılması		
Gereksinim_1			TD.03	Özellik güncelleme bağlı iş ortağı depoları yetki tanımı yapılması		
Gereksinim_1			TD.04	Özellik güncelleme servis yetki tanımı yapılması		
Gereksinim_1			TD.05	Özellik güncelleme müşteriye verilebilir yetki tanımı yapılması		
Gereksinim_2			TD.06	Sisteme başarılı giriş yapılabilmesi		
Gereksinim_2			TD.07	Sisteme yanlış girişin üçten fazla yapıldığı durum kontrolü		
Gereksinim_5			TD.08	Cihaz değişimi başarılı şekilde yapılması		
Gereksinim_6			TD.09	Cihaz bilgilerinin stoka yanlış girildiği durum kontrolü		
Gereksinim_7			TD.10	Yanlış cihazın değişim işleminde hata alınması		
Gereksinim_8			TD.11	Cihaz katalog yönetiminde formatı hatalı veri girişi yapılması		
Gereksinim_9			TD.12	Cihaz özelliklerinin başarılı güncellenmesi		
Gereksinim_10			TD.13	Yanlış depolara cihaz sevk işlemi		
Gereksinim_11			TD.14	Yanlış cihazların depolara sevk işlemi		

Test durumları oluşturulurken gereksinimler dikkate alınarak oluşturulur. Gereksinimleri karşılayan test durumları ürün risklerine göre önceliklendirilir. Öncelik seviyesi:

- Kritik
- Yüksek
- Orta
- Düşük

Bir gereksinim için birden fazla test durumu yazılabilir ya da bir test durumu birden fazla gereksinimi karşılayabilir. Bu sisteme ait gereksinimlere göre değişebilir. Çizelge 8.13 şablonu CKSY projesinde kullanılmıştır.

Spesifik test verilerinin tanımlanması (SP 1.3)

Belirlenen test senaryoları gerçekleştirilmesi için senaryolar için veri oluşturulması gerçekleştirildiği bu uygulamada test verileri aşağıdaki kriterler dikkate alınarak oluşturulmuştur:

- Müşteri tipine göre oluşturulan müşteri bilgileri
- Müşteriliklere göre oluşturulan hizmetler
- Hizmet bilgileri
- Kampanya bilgileri
- Cihaz bilgileri
- Stok bilgileri
- Depo bilgileri
- Kullanıcı bilgileri ile rol ve yetkileri

Gereksinimlerle yatay izlenebilirliğin korunması (SP 1.4)

Gereksinim izlenebilirlik için aşağıdaki şablon kullanılmıştır. Dokümanda ayrı ayrı sheetlerde kullanılan tablolar aşağıda verilmiştir.

Çizelge 8.13. Gereksinim izlenebilirlik matrisi

H7 İlişkili GTD Dokümanı

GEREKSİNİM İZLENEBİLİRLİK MATRİSİ

Proje Adı	Cihaz Kiralama ve Stok yönetimi projesi
Proje Yönetici Adı	
Proje Açıklama	Kurum tarafından cihaz kiralama modelinin, cihaz satış modeline ek yeni bir iş modeli olarak müşterilere sunulması amaçlanmaktadır. Bu iş kapsamında başlatılan proje ile hem bu yeni modelin satış sistemine entegre edilmesini sağlayacak hem de cihazların stok hareketlerinin

Gereksinimler							**Analiz**	**Tasarım**		**Gelişt**	**Test**			
Gereksinim No	**Gereksinim tipi**	**Teknik Varsayım(lar) ve / veya Müşteri İhtiyaçları (Req Alt Türü)**	**Fonksiyonel Gereksinim Açıklaması**	**Öncelik**	**AS-IS/TO-BE**	**Genel durum**	**İlişkili GTD Dokümanı**	**Mimari/Tasarım Dokümanı**	**Teknik Maddeler**	**Ortam bilgisi**	**Test Durum No**	**Test Durum Adı**	**Doğrulama**	**Ek Yorumlar**
Gereksinim_1	Fonksiyonel		• ozellikGuncelleme: Cihaz özellik güncelleme ekran yetkisidir. Bu yetkiye sahip kullanıcılar cihaz özellik güncelleme ekranında kendi depolarındaki cihazları sorgulayabilecektir.	Kritik	To be	analiz OK	CKSY_Projesi_GTD_v3	CKSY_projesi_ÇTD_v3	Tasarım_01	Test	TD_01	özellik güncelleme yetki tanımı yapılması	GEÇTİ	
Gereksinim_1	Fonksiyonel		ozellikGuncellemeTumDepolar: Kendi deposunun haricinde tüm depolardaki cihazların görüntülenmesini sağlayan yetkidir.	Kritik	To be	analiz OK	CKSY_Projesi_GTD_v3	CKSY_projesi_ÇTD_v3	Tasarım_01	Test	TD_02	özellik güncelleme tüm depolar yetki tanımı yapılması	GEÇTİ	
Gereksinim_1	Fonksiyonel		ozellikGuncellemeBagliIsOrtagiDepolari: Kendi deposunun haricinde kullanıcının illerinde yer alan iş ortağı depolarının görüntülenmesini sağlayan yetkidir.	Kritik	To be	analiz OK	CKSY_Projesi_GTD_v3	CKSY_projesi_ÇTD_v3	Tasarım_01	Test	TD_03	özellik güncelleme bağlı iş ortağı depoları yetki tanımı yapılması	GEÇTİ	
Gereksinim_1	Fonksiyonel		ozellikGuncellemeServis: Cihazların servis özelliğinin güncellenmesine sağlayan yetkidir	Kritik	To be	analiz OK	CKSY_Projesi_GTD_v3	CKSY_projesi_ÇTD_v3	Tasarım_01	Test	TD_04	özellik güncelleme servis yetki tanımı yapılması	GEÇTİ	
Gereksinim_1	Fonksiyonel		ozellikGuncellemeMusteriyeVerilebilir: Cihazların müşteriye verilebilir özelliğinin güncellenmesini sağlayan yetkidir.	Kritik	To be	analiz OK	CKSY_Projesi_GTD_v3	CKSY_projesi_ÇTD_v3	Tasarım_01	Test	TD_05	özellik güncelleme müşteriye verilebilir yetki tanımı yapılması	GEÇTİ	
Gereksinim_2	Fonksiyonel		CKSY sistemine girişte kullanıcı adı,şifre ve captcha alanları kullanılmalıdır.	Kritik	as-is	analiz OK	CKSY_Projesi_GTD_v3	CKSY_projesi_ÇTD_v3	Tasarım_02	Test	TD_06	Sisteme başarılı giriş yapılabilmesi	GEÇTİ	
Gereksinim_2	Fonksiyonel		Sisteme girişte yanlış girme hakkı 3 olmalıdır. Üçten fazla girildiğinde uyarı verilmelidir.	Normal	as-is	analiz OK	CKSY_Projesi_GTD_v3	CKSY_projesi_ÇTD_v3	Tasarım_02	Test	TD_07	Sisteme yanlış girişin üçten fazla yapıldığı durum kontrolü	GEÇTİ	

Çizelge 8.14. Gereksinim izlenebilirlik matrisi alan adları

Gereksinim No	Dokumandaki numaralandırılmış maddenin numarası girilir
Gereksinim Tipi	Gereksinim Fonksiyonel ya da Fonksiyonel olmadığı bilgisi girilir
Teknik Varsayım (lar) ve / veya Müşteri İhtiyaçları (Req Alt Türü)	Teknik varsayımlar varsa onları belirtilir
Fonksiyonel Gereksinim Açıklaması	Fonksiyonel gereksinimler girilir
Öncelik	Gereksinimlerin öncelik seviyeleri belirtilir. Örnek: Kritik, Normal vb.
AS-IS/TO-BE	Ör: TO-BE
Genel durum	Gereksinim maddelerinin durumu belirtilir. Ör: Analiz tamamlandı
İlişkili GTD Dokümanı	GTD dokumanın ismi versiyonuyla birlikte girilir
Mimari/Tasarım Dokümanı	Tasarım dokuman ismi girilir
Teknik Maddeler	Tasarım dokümanındaki gereksinimlerin numaraları girilir.
Ortam bilgisi	Sistem bazlı ortam bilgileri girilir
Test Durum No	Test durum numaraları girilir
Test Durum Adı	Test durum adları girilir
Doğrulama	Test durumları doğrulama bilgileri girilir. GEÇTİ /KALDI vb.
Ek Yorumlar	Yorumların girileceği alandır

Çizelge 8.15. Gereksinim izlenebilirlik matrisi değişiklik kayıtları

Versiyon	**Değişiklik Tanımı**	**Hazırlayan**	**Tarih**
v1	Gereksinim maddeleri eklendi		

Test Uygulamasının Gerçekleştirilmesi (SG 2)

Test yürütme programı aşağıdaki uygulamalar ile oluşturulmuştur.

Test prosedürlerinin geliştirilmesi ve önceliklendirilmesi (SP 2.1)

Test ortamının oluşturulması, testlerin koşturulması ve sonuçlarının değerlendirilmesi için ayrıntılı yöntemler, test durumları, açıklamalar listesi ve test planı temel alınarak test prosedürleri geliştirilir. Aşağıda tanımlanan prosedür adımları bulunmaktadır:

1. Hatasız olarak derlenmiş kodu olan yazılım alınır.
2. Yazılımın teste hazır olup olmadığı kontrol listesi kullanılarak doğrulanır.
3. Test ortamının hazır olup olmadığı kontrol listesi kullanılarak doğrulanır.
4. Yazılım çalıştırılır.
5. Yardımcı test yazılımları çalıştırılır.
6. Sırası ile test durumları koşturulmaya başlanır.

Özel test verileri oluşturulması (SP 2.2)

Test analizi ve tasarım aktivitesi sırasında belirlenen özelliklere göre test verileri oluşturulur.

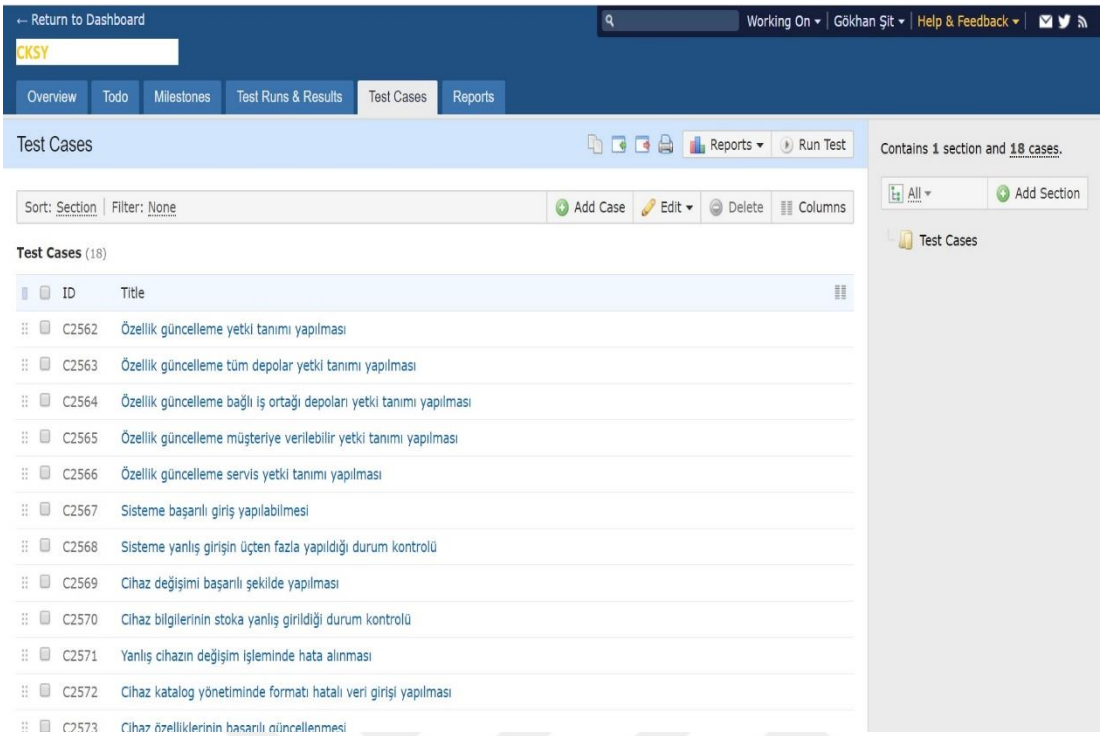
Giriş testi prosedürünün belirtilmesi (SP 2.3)

Test nesnesi ayrıntılı ve ileri test için hazır olup olmadığını test yürütmesinin başında karar vermek için güven veya duman testi olarak adlandırılan test kullanılmıştır. Giriş testinin parçası olacak kontroller aşağıdakileri içerecek şekilde oluşturulur:

- Gerekli tüm önemli işlevlerin erişilebilir olması
- Test edilecek diğer bileşenlerle veya sistemlerle ara yüzlerin çalışıyor olması
- Mevcut fonksiyonlar için dokümantasyon tamamlanması
- Ör. Test sürüm notu, kullanım kılavuzu, montaj kılavuzu

Test yürütme programı geliştirilmesi (SP 2.4)

Test durumları oluşturulduktan sonra TestRail tool'u kullanılarak takibi sağlanmıştır. Şekil 8.3' de TestRailde test durumlarının ekran görüntüsü verilmiştir.



Şekil 8.3. TestRail toolunda oluşturulmuş test durumları ve diğer menüler

Test Yürütülmesinin Gerçekleştirilmesi (SG 3)

Giriş testi yapılması (SP 3.1)

Yazılım teslim edildikten sonra güven testi için duman (smoke) testi olarak adlandırılan testler gerçekleştirilir.

Şekil 8.4. Smoke test süreci

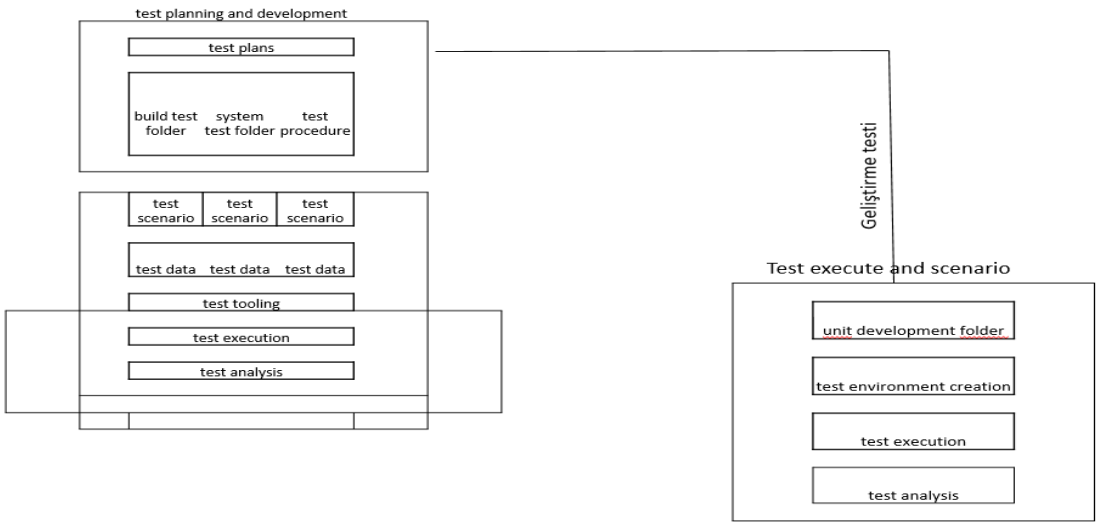
Smoke testi kapsamında testi yapılan durumları Çizelge 8.17'de verilmiştir.

Çizelge 8.16. Smoke test durumları

Test Durum No	**Test Durum Adı**
1	Sisteme giriş yapılması
2	Cihaz yönetimi yapılabilmesi
3	Depo yönetimi yapılabilmesi
4	Kullanıcı rol ve yetki tanımı yapılabilmesi
5	Cihaz bilgilerinin güncellenebilmesi
6	Cihaz durum güncellemesi yapılabilmesi

Test durumlarının yürütülmesi (SP 3.2)

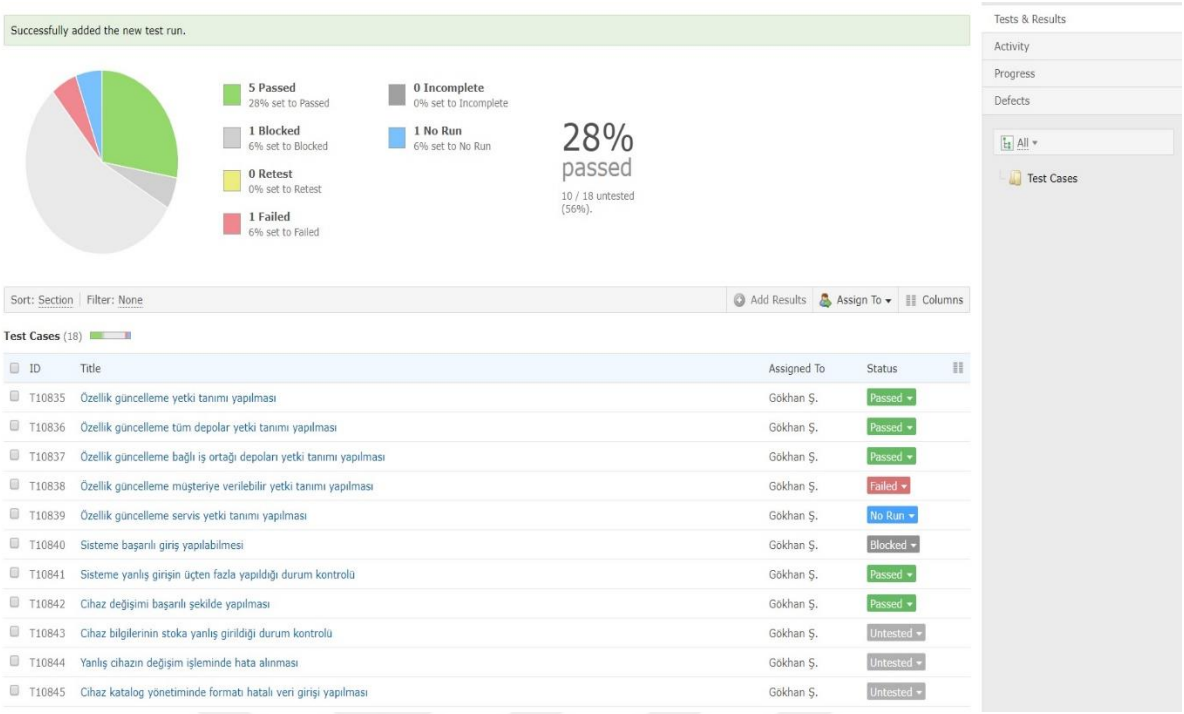
Test durumları test durum tablosu şablonu kullanılarak ve TestRail tool'u yardımıyla takip edilmiştir. Test yürütülmesi uygulama aşamasında aşağıdaki şekilde verilmiş süreç kullanılabilir.



Şekil 8.5. Test yürütme programı

Test durum raporları (SP 3.3)

Ekler bölümünde test kapanış raporları bulunmaktadır. Ayrıca TestRail toolunda her bir test durumu için durumların takibi Şekil 8.6'da olduğu gibi sağlanmıştır.



Şekil 8.6. TestRail toolunda test durum statüleri ve raporu

Test loğlarının yazılması (SP 3.4)

Test loğları mevcutta kullanılan TestRail tollu ile sağlanmıştır. Bu tool ile oluşturulmuş test durumlarının statüleri Resim 8.2'de verildiği gibi, açılmış hatalar, hataların statüleri, güncelleyen, ekleyen kullanıcı bilgileri, testin gerçekleştirildiği zaman vb. bilgiler tutulmaktadır.

Test durumlarının kapanışının yönetilmesi (SG 4)

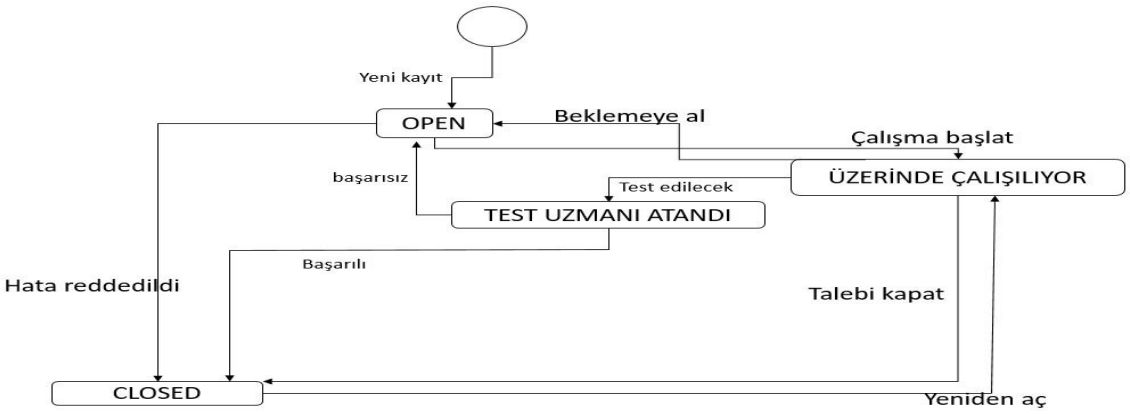
Bu süreç alanı aşağıdaki uygulamalar ile sağlanmıştır.

Yapılandırma kontrol panosunda olayların düzenlenmesine karar verilmesi (SP 4.1)

TestRail ile bulunan hatalar, test durumları öncelik ve önem derecelerine göre gözden geçirilmiştir. Hata durumları ve statülerine göre kararlar verilerek ilerlenir.

Test durumlarında karşılaşılan hataları düzeltmek için uygun eylemin gerçekleştirilmesi (SP 4.2)

CKSY projesi kapsamında hata yönetimi süreci aşağıdaki gibi gerçekleştirilmiştir:
Şekil 8.7'de verilen hata yönetim sürecinde olduğu gibi test sorumlusu hata tespit ettiğinde hatanın önem derecesini belirleyerek JIRA (Atlassian tarafından geliştirilen ve hata takibi ve çevik proje yönetimi sağlayan hata izleme aracıdır.) üzerinde defect kaydını oluşturur. Hatanın kime açılacağı test sorumlusu tarafından belirlenir. Testlerin durdurulması halinde yeniden test için testlere ara verilir. Durdurma gerektirmeyen yani etkilenmeyen alanların testlerine devam edilebilir. Hatanın atandığı kişi çalışmayı başlatarak analist ile ya da tek başına analizini yapar ve hatanın giderilmesi yönünde düzeltmeyi yapar ve defect statüsünü üzerinde çalışılıyor durumuna çeker. Hata düzeltildiyse ve hata olduğu kabul edildiyse JIRA üzerinden hatanın durumunu Test uzmanı atandı olarak günceller ve yeniden test edilmesi için testçiye atama yapar. Hata düzeltilmiş ve test başarılıysa defect kaydı kapatılır. Geliştirmeyi yapan hata olmadığını düşünüyorsa hatayı reddeder ve defect kapatılır. Hata tekrar testinden başarısız olursa yeniden aç olarak güncellenir.



Şekil 8.7. Test defect yönetimi

Test hata durumunun izlenmesi (SP 4.3)

Bu uygulamada test durumlarında karşılaşılan hatalar için kullanılan rapor şablonları izleme hataları uygulamasında verilmiştir.

Test Ortamı

Test ortamı gereksinimlerini belirlemek, test ortamını uygulamak, test ortamını yönetmek ve kontrol etmek gibi aktivitelere hitap eder. Test ortamının yönetimi ve kontrolü ayrıca yapılandırma yönetimi ve kullanılabilirliği sağlama gibi hususları içerir.

Test Ortamı Süreç Alanı için Özel Hedefler(SG) ve Özel Uygulamalar(SP)

Test Ortamı Gereksinimlerinin Geliştirilmesi (SG 1)

Bu süreç alanı aşağıdaki uygulamalar ile gerçekleştirilmiştir.

Test ortamı ihtiyacı oluşması (SP 1.1)

Test ortamı gereksinimleri şunları içerir:

- Ağ bileşenleri
- Yazılım bileşenleri. Örneğin işletim sistemleri
- Simülatörler, taslaklar ve sürücüler
- Destekleyici belgeler, Ör. Kullanıcı kılavuzları, teknik kılavuzlar ve kurulum kılavuzları
- Ara yüz bileşenleri veya ürünleri
- Sürücü geliştirme araçları
- Çoklu test ortamları için gereksinimler
- Test veri tabanları
- Test veri üreteçleri
- Veri arşivini test etme ve geri yükleme olanakları

Ortamlar;

- Geliştirme ortamı
- Test ortamı
- Canlı ortamı

Test ortamı gereksinimlerinin geliştirilmesi (SP 1.2)

Yukarıda belirlenen gereksinimlerin önceliklendirilmesi sağlanır.

Test ortamı gereksinimlerinin analizi (SP 1.3)

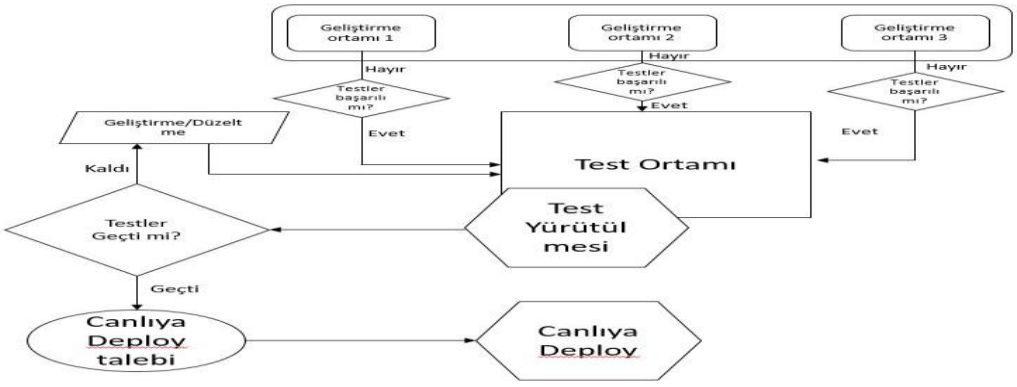
Canlı ortam ile uyumlu bir test ortamı için analiz yapılarak ortam gereksinimleri değerlendirilir.

Test Ortamı Uygulamasının Gerçekleştirilmesi (SG 2)

Bu süreç alanı aşağıdaki uygulamalar ile gerçekleştirilmiştir.

Test ortamının kurulması (SP 2.1)

Test ortamları analiz aşamasından canlı ortam ile aynı seviyede olan birden fazla ortam olacak şekilde kurulmuştur.



Şekil 8.8. CKSY sistemi test ortamı

Genel test verileri oluşturulması (SP 2.2)

Kurulan test ortamlarının testleri için canlıda mevcutta olan verilerin özellikleri test ortamında sağlanmıştır.

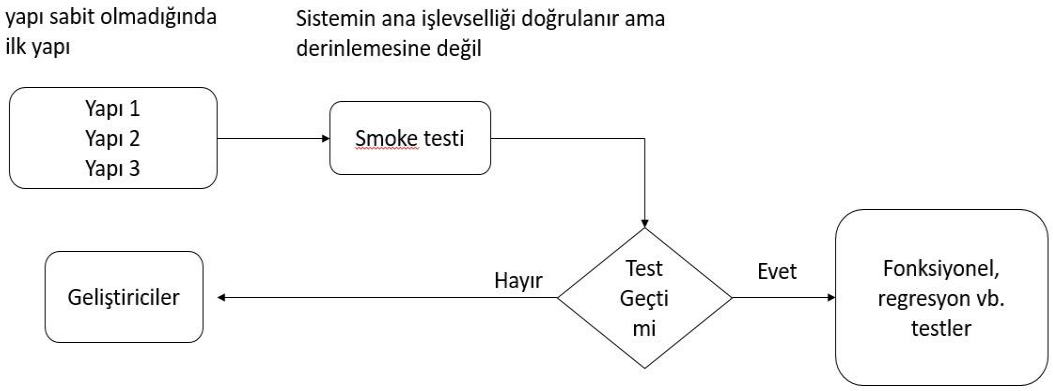
Test ortamı giriş testi prosedürünün belirtilmesi (SP 2.3)

Test ortamı giriş testi (güven testi) yapılması için prosedür belirlenmiştir.

1. Hatasız olarak derlenmiş kodu olan yazılım alınır.
2. Yazılımın teste hazır olup olmadığı kontrol listesi kullanılarak doğrulanır.
3. Test ortamının hazır olup olmadığı kontrol listesi kullanılarak doğrulanır.
4. Yazılım çalıştırılır.
5. Yardımcı test yazılımları çalıştırılır.
6. Sırası ile test durumları koşturulmaya başlanır.

Test ortamı giriş testinin gerçekleştirilmesi (SP 2.4)

Test ortamı giriş testi (güven testi) gerçekleştirilmiştir.



Şekil 8.9. Test ortamı için yapılan smoke test süreci

Test Ortamlarının Yönetilmesi ve Kontrol Edilmesi (SG 3)

Test ortamlarının kesintisiz çalışması için aşağıdaki uygulamalar uygulanmıştır.

Sistem yönetiminin gerçekleştirilmesi (SP 3.1)

Geliştirme ortamlarında geliştirmeler yapılmadan test ortamı ile uyumlu olması için test ortamı pull edilir. Bu şekilde geliştirmeler yapıldıktan sonra geliştirmenin yapıldığı yerler test edilirken sorunlar oluşmadan sağlıklı testler gerçekleştirilir. Test ortamı içinde canlı ortam ile uyumlu olması gerekir. Canlıya çıkacak tüm geliştirmelerin sağlıklı testi için gereklidir. Bu şekilde test ortam yönetimi CKSY projesi kapsamında konfigürasyon yöneticisi ve geliştiriciler ile sağlanmıştır.

Test veri yönetiminin gerçekleştirilmesi (SP 3.2)

Test verileri aşağıdaki tabloda gereksinim durumlarına göre ilgili sistemlerden yeni oluşturularak ya da veri tabanında SQL sorguları yardımıyla gerçekleştirildi. Bir sonraki seviye 4 te test veri yönetimi için otomatik olarak test veri yönetimini gerçekleştiren araçlar ile veri yönetiminin gerçekleştirilmesi hedeflenmektedir.

Çizelge 8.17. Test veri yönetimi

Test Hedefleri	**Durum(Yapıldı/Yapılmadı)**
CKSY sistemini kullanıma uygun olarak doğrulamak.	Yapıldı
Operasyonda meydana gelebilecek hataları önlemek.	Yapıldı
Standartlara uygunluğu doğrulamak.	Yapıldı
Ürün kalitesi ile ilgili görünürlük sağlamak.	Yapıldı
Testler için gerekli test ortamını tanımlamak.	Yapıldı
Testler için gerekli verilerin hazırlanma şekli tanımlamak.	Yapıldı
Testler süresince uygulanacak teste giriş ve testten çıkış ölçütleri, testlerin geçme ve kalma ölçütleri tanımlamak.	Yapıldı
Test faaliyetlerinden sorumlu olacak personellere rol ve sorumlulukları hakkında bilgi vermek.	Yapıldı
Test yürütme süresini kısaltmak.	Yapıldı

Test ortamlarının kullanılabilirliğinin ve kullanımının koordine edilmesi (SP 3.3)

Canlıdan gelen hataların geliştirilmesi ya da yeni geliştirmeler için her bir geliştirici kendi geliştirme yaptıkları ortamlarda bu model ile kullanılabilirlik yakalanmış ve ekip tarafından benimsendiği için kullanımın koordine edilmesi geliştiriciler tarafından sağlandığından kullanılabilirlik ve yönetilebilirlik bu uygulama ile tarif edildiği şekilde gerçekleştirilmiştir.

Test ortamı olaylarının rapor edilmesi ve yönetilmesi (SP 3.4)

Test ortamı testlerinde karşılaşılan durumlar için kullanılan test durum raporları tabloları izleme hataları bölümünde verilmiştir.

8.1.2. Seviye 2 düzeyinin değerlendirilmesi ve olgunluk seviyesinin belirlenmesi

Mevcut süreç belirlenip TMMi modeli ile seviye 2 düzeyine geçiş süreci yukarıdaki özel hedefler için uygulanan özel uygulamalar sonucunda tamamlanarak seviye 2 değerlendirmesi yapılmıştır. Özel hedeflerde olması gereken iş ürünleri uygulama bölümlerinde verilmiştir. Seviye 2 düzeyine geçişin tamamlandığı ve seviye 2 hedefinin karşılanıp karşılanmadığı yine ekip tarafından önerilen yöntem kullanılarak sağlaması gerçekleştirildi. Seviye 2 değerlendirmesi için verilen yanıtlar Çizelge 8.18'de verilmiştir.

Çizelge 8.18. TMMi 2. Seviye değerlendirme için verilen ortalama puanlar ve başarı puanı

TMMi 2.Seviye	Verilen Puanların Ort.
1.Şirketinizde projeler ya da talepler için test politikası ve test stratejileri tanımlanıyor.	4,71
2.Test performans göstergeleri(test efor ve maliyeti, test süresi, bulunan hata sayısı, hata tespit yüzdesi, test kapsamı vb.) tanımlanıyor.	4,28
3.Test planlaması ve risk değerlendirmesi yapılıyor.	4,28
4.Test yaklaşımları (test edilecek öğeler, giriş ve çıkış kriterleri, durdurma ve başlatma kriterleri vb.) tanımlanıyor.	4,14
5.Plana dayalı test ilerleme süreci izleniyor ve raporlamalar yapılıyor.	4,28
6.Her test seviyesi(Birim, Entegrasyon, Sistem, Kabul testleri) açık bir şekilde tanımlanarak test hedefleri oluşturuluyor.	4
7.Test tasarım teknikleri(Kara kutu, Beyaz kutu, Deneyim temelli test teknikleri vb.) kullanılıyor.	4
8.Test durumları tanımlanıyor ve öncelik belirleniyor.	4,57
9.Hata yönetim süreci bulunuyor, test sırasında karşılaşılan hatalar raporlanıyor.	4,28
10.Yönetilebilir ve kontrol edilebilir test ortamı bulunuyor.	4,42
ORTALAMA	**4,296**
BAŞARI	**85,92**

Proje kapsamında çalışanlar tarafından sorulara verilen puanların ortalama değerleri ve Seviye 2 için başarı yüzdesi %85,92 olarak çıktığı görülmektedir. Bu sonuca göre CKSY

projesi üzerinde test olgunluk seviyesi Seviye 2 olarak belirlenmiştir. Bundan sonraki bölümde Seviye 3 düzeyine geçiş ve seviye 3 değerlendirmesi için önerilen yöntemin uygulanışı verilecektir.

Ayrıca seviye 2 geçiş sürecinde yapılanlar ve elde edilenlerin özeti aşağıdaki gibidir:

- Bu seviyede test kontrol edilebilen bir süreç haline gelmiştir.
- Süreç disiplini sağlanarak belirli yük altında uygulamaların çalıştığından emin olunmuştur. Bununla birlikte testin, kodlamadan sonra yapılan bir süreç olduğu çalışanlar tarafından benimsenmiştir.
- Test planları oluşturularak içinde test yaklaşımı tanımlanmış ve bu yaklaşım proje risk değerlendirmesine uygun olarak hazırlanmıştır.
- Proje riskini belirlemek için risk yönetim teknikleri, gereksinimlere bakarak tanımlanmıştır.
- Testte ne istendiği, ne zaman ve nasıl istendiği gibi bilgiler test planında tanımlanarak paydaşlarla birlikte gözden geçirilerek yapılmıştır. Testi izleme, kontrol etme ve plana uygun hareket edilmesi sağlanmış ve sapma olma olasılığına karşı düzeltici eylemler gerçekleştirilmiştir.
- Test tasarım teknikleri belirlenerek test senaryoları oluşturulmuştur.
- Bu aşamada bileşen, entegrasyon, kabul ve sistem testi seviyeleri tanımlanmıştır.
- Tanımlanan test seviyeleri için test hedefleri tanımlanmış ve test ve kod uygulaması süreci ayrı işletilecek duruma gelmiştir.

8.2. Seviye 2 den Seviye 3 e Geçiş Süreci

8.2.1. Seviye 3 süreç alanları ve uygulamaları

CKSY sistemi üzerinde test olgunluk seviyesi belirlendikten sonra test olgunluk seviyesi 3 düzeyine çıkabilmek için TMMi modelinde verilmiş seviye 3'e ait süreç alanları ve bu süreç alanlarına ait özel hedefler ve özel uygulamalar gerçekleştirilmiştir.

Seviye 3 süreç alanları şunlardır:

- Test Organizasyonu
- Test Eğitim Programı
- Test Yaşam Döngüsü ve Entegrasyonu
- Fonksiyonel Olmayan Test
- Eş Gözden Geçirme

Test Organizasyonu

Test organizasyonu süreç alanı, işleyişi (görevler, sorumluluklar, raporlama yapısı) ve genel organizasyondaki bir test grubunun konumunu tanımlar. Test rolleri, fonksiyonlar ve kariyer yolları testin profesyonel bir disiplin olarak kabul edilmesini desteklemek için tanımlanmıştır. Test organizasyonunda test sürecinin iyileştirilmesi, mevcut test sürecinin değerlendirilmesini ve olası test iyileştirmelerini tanımlamak için öğrenilen dersleri, geliştirmeleri uygulamak ve projelerin test faaliyetlerinde bunları dağıtmaktır. Test organizasyonu süreç alanında amaç, testten sorumlu bir grup yetenekli insanın tanımlanması ve bu grubun organize edilmesidir. Test grubuna ek olarak, test grubu kuruluşun test sürecine ve test süreci varlıklarına, kuruluşun mevcut test sürecinin ve test süreci varlıklarının güçlü ve zayıf yönlerini tam olarak anlamaya dayalı iyileştirmeleri de yönetir.

Test Organizasyon Süreç Alanı için Özel Hedefler(SG) ve Özel Uygulamalar(SP)

Test Organizasyonu Kurma (SG 1)

Projelerde ve organizasyonda test uygulamalarını destekleyen bir test organizasyonu aşağıdaki uygulamalar kullanılarak tanımlanmıştır.

Test organizasyonu tanımlama (SP 1.1)

Test organizasyonu için test politikası, organizasyonun görevleri ve sorumlulukları tanımlandı. Test Politikası: "Olgunlaşmış test kültürü ve süreçleri oluşturmak" olarak belirlenmiştir. Test ekibi geliştirilmiş olan yazılım çözümlerinin ayrıntılı bir şekilde testlerini yapmaktan sorumlu olan roldür. Test senaryolarının hazırlanması ve gerçek hayatta karşılaşılacak durumların olabildiğince her türlüsünü canlandırmak, yazılımın kabiliyetlerini, performansını, kalitesini ve beklentilere uygunluğunu test etmekten sorumludur. Görev ve sorumlulukları:

- Gereksinimlerden üretilmiş tasarıma uygun test senaryolarının yazılması.
- Geliştirilmiş yazılım çözümlerinin senaryolara uygun şekilde test edilmesi
- Çözümlerin olası kullanıcı davranışlarına takılmaması için uygun hale getirilmesi
- Çözümlerin kullanıcı kabulüne hazır hale getirilmesi
- Çözümlerin performans olası dar boğaz bakımından test edilmesi
- Test senaryolarının dokümantasyonunun oluşturulması
- Test sonuçlarının dokümantasyonunun oluşturulması
- Test araç ve metot gereksinimlerinin belirlenmesi ve karşılanmasının sağlanması
- Test süreç ve metotların diğer çözüm geliştirme süreçleri ile entegre edilmesi
- Direktörün verdiği görevlerin yerine getirmek.

Test ekibi;

- Bir test uzmanı

- Bir test personeli
- Gereksinim ve tasarım için sistem uzmanı(analist)
- Test yönetimi için direktör

Test Uzmanları için Test Fonksiyonları Kurma (SG 2)

Test görevleriyle birlikte test fonksiyonları oluşturulduğu ve test uzmanlarına atandığı bu süreç alanı için aşağıdaki uygulamalar gerçekleştirilmiştir.

Test fonksiyonlarını tanımlama (SP 2.1)

Test fonksiyonları:

- Test yöneticisi
- Test ekibi lideri
- Test tasarımcısı
- Test mühendisi

Uzmanlaşmış alanlar için test fonksiyonları:

- Test otomasyon mimarisi
- Test otomasyon mühendisi
- Performans testi mühendisi
- Kullanılabilirlik test mühendisi
- Test süreci iyileştirme lideri

İş tanımlarını geliştirme (SP 2.2)

İş tanımlarını tanımlanırken ve geliştirilirken kuruluşun standart test süreci girdi olarak kullanıldı. Tanımlanan test fonksiyonları için iş tanımlar geliştirilmiştir.

İş açıklamaları:

- Test fonksiyonunun adı
- Kısa Açıklama
- Maaş skalası
- Nitelikler
- Gerçekleştirilecek tipik görevler
- Sorumluluklar ve yetkililer
- Gerekli bilgi ve beceri
- Eğitim gereksinimleri
- Takip edilecek eğitim modülleri

İş tanımları kuruluşun İnsan Kaynakları Yönetimi çerçevesi dâhil edilerek oluşturuldu. Test aktivitelerinin ve sorumluluklarının kapsadığı test dışı uzmanların iş kategorileri:

- Yazılım geliştirici
- Sistem mühendisi
- Sistem entegratörü
- Kullanıcı temsilcisi(Müşteri)

Personeli test fonksiyonlarına atama (SP 2.3)

Test organizasyonunda üyeleri için tanımlanan test fonksiyonları atanır.

Test Süreci İyileştirmelerini Belirleme, Planlama ve Uygulama (SG 3)

Kuruluşun test sürecini değerlendirme (SP 3.1)

Kuruluşun test süreci bu tezde verilen olgunluk seviye belirleme modeli kullanılarak belirlendi. Seviye sonucu seviye 1 olarak çıkan bu test olgunluk seviyesindeki durumlar şu

şekildedir: Önceden kurgulanmamış ve çoğunlukla debugging'den oluşan anlık test yöntemleri uygularlar. Bulgular tekrar edilemez ve herhangi bir kalite standardı veya süreci yoktur. Genel başarı sadece kişilerin yetkinlikleri ve inisiyatiflerine bağlı şekilde elde edilebilir. Uygulanan test aktivitelerinin temel amacı ise sadece uygulamaların "çok büyük hatalar içermeden" çalışmasıdır.

Kuruluşun test süreci iyileştirmelerini tanımlama (SP 3.2)

Test süreci iyileştirmelerinin önceliğini belirleyen faktörler önceki bölümlerde verilmiş olan test süreci iyileştirme başarı faktörleri:

- İyileştirme Süreci için Net, Ölçülebilir ve Gerçekçi Hedefler
- Yönetim taahhüdü ve mevcut sponsorluk
- İyileştirme İhtiyacı
- Resmi Proje Olarak Düzenlenen Test İyileştirme
- İyileştirmeye katılan katılımcılara yeterli zamanın verilmesi
- Organizasyonun olgunluk seviyesine göre hedeflerin belirlenmesi
- İyileştirmeler için zaman çerçeveleri ve geri besleme döngüleri uzunluğu tanımlanmıştır.
- Direnç seviyesi, önceki iyileştirme çabalarının başarısına veya başarısızlığına bağlı olacaktır.
- Hâlihazırda mevcut olan mevcut uygulamaları kullanılır; değişmek uğruna değişilmemelidir. Kullanılabilir bir şey kullanılmıyorsa, önce nedenleri araştırılır.
- Örneğin, belirli bilgi ve beceriler için gerekli olan harici danışmanlar dâhil edilir, ancak bunların iyileştirme projesi için tüm sorumluluğu üstlenmelerine izin verilmemelidir.
- Koruma Tutarlılığı
- Her döngü için net, ölçülebilir ve gerçekçi iyileştirme hedefleri.

- Değişim yönetimi sürecindeki tüm adımların kontrolü ve izlenmesi
- Geliştirmeleri tanımlarken ve uygularken ilgili test uzmanları
- Sorunların test disiplini dışında yer aldığı diğer paydaşlar (örneğin, özelliklerin kalitesi, değişim ve yaygınlaştırma yönetim süreçleri)
- Birlikte iyi çalışan ve değişim/vizyona giren kararlı proje ekibi
- Test iyileştirmelerini desteklemek ve/veya etkinleştirmek için araçlar
- Katılan insanların becerileri. Bu sadece genel olarak test değil, aynı zamanda kullanılacak iyileştirme yaklaşımı için iyileştirme süreci ve becerileri ile ilgili alanları (örneğin, belirli model, analiz teknikleri) kapsar.
- Öğrenme stilleri, kişilik türleri ve kabul edilen tutumlar gibi insan faktörleri
- Zorunlu olabilecek dış standartların farkındalığı
- İyileştirme stratejisinin çeşitli bileşenlerinin hizalanmış ve genel bir çerçevenin bir parçası olmasını sağlamak için genel süreç ve terminolojinin ön planda tanımlanması.
- Etkilenen tüm paydaşlarla (örneğin, yazılım süreci iyileştirme görevlileri, kalite güvencesi ve insan kaynakları departmanları) oluşturulmuş ilişkiler.
- İç onay ve / veya düzenleyici süreçlere itaat.
- Diğer iyileştirme girişimleri ile uyum.

Test süreci iyileştirme planlaması (SP 3.3)

Test süreci iyileştirme planının öğeleri:

- Test süreci iyileştirme hedefleri: Tanımlanan eksikliklerin giderilmesi, belirli bir test olgunluk seviyesine ulaşma, otomasyon sürecine erişebilme.
- Test süreci iyileştirme organizasyon yapısı: Test ekibi oluşturuldu. Test lideri pozisyonu için test uzmanı ile birlikte bir test personeli ve yönetimi için direktör. Gereksinimlerin gözden geçirilmesi ve daha sonra test durumlarının oluşturulması

ve bunların gözden geçirilmesi bu ekip tarafından gerçekleştirildi. Test durumları test otomasyon süreci için analiz edilir ve buna göre test durumları geliştirilir.

- İzleme ve kontrol için prosedürler: İzleme ve kontrol için her hafta belirlenen gün ve ekiplerce toplantılar yapılır.

- Test süreci iyileştirmeleri için pilot uygulama ve stratejiler: Test süreci iyileştirmesi bu tez kapsamında CKYS sistemi ele alındığı için pilot uygulama olarak bu sistem seçilir.

- Sorumluluklar ve yetkililer: Test uzmanı ve test personeli ile direktör olarak yetkililer ve daha önce tanımlanan sorumluluklardır.

- Kaynaklar ve çizelgeler: Test personeli ve test uzmanı olarak atanan test lideri ve test toolları, oluşturulan dokümanlar(gereksinim tanımlama dokümanı, gereksinim izlenebilirlik matrisi, test planı, test durum dokümanları vb.) kaynaklar olarak belirlenir.

- Test süreci iyileştirme planı ile ilişkili risk: Test süreci iyileştirilmesi sürecinde mevcutta devam eden işlerde oluşacak aksaklıklar, yapılma ihtimali olan fazla mesailer risk olarak belirlenir.

Test süreci iyileştirmelerini uygulama (SP 3.4)

Test süreci iyileştirmeleri tanımlanan faktörler ve planlara göre gerçekleştirildi. Uygulama olarak TMMi modeli uygulanmıştır.

Test Eğitim Programı

Süreç alanı test eğitim programı, organizasyonel test eğitim planının ve test eğitim kabiliyetinin oluşturulmasına yöneliktir. Planlanan test eğitiminin gerçek teslimini de ele alır. Projeye özel eğitim ihtiyaçları bu süreç alanının bir parçası değildir. Bunlar test planlaması süreç alanında ele alınmaktadır. Test eğitim programı süreç alanının amacı, insanlara bilgi ve beceri geliştirmeyi kolaylaştıran bir eğitim programı geliştirmektir. Böylece test görevleri ve rolleri etkin ve verimli bir şekilde gerçekleştirilebilir.

Test Eğitim Programı Süreç Alanı için Özel Hedefler(SG) ve Özel Uygulamalar(SP)

Örgütsel Test Eğitimi Yeteneği Kurmak (SG 1)

Kuruluşun test rollerini destekleyen bir eğitim yeteneği oluşturulur.

Stratejik test eğitimi ihtiyaçlarını belirleme (SP 1.1)

Test eğitimi ihtiyaçlarının kategorileri:

- Test mühendisliği ve süreci (ör. Organizasyonel standart test süreci, test prensipleri, test yaşam döngüsü, statik test teknikleri, dinamik test teknikleri, test araçları ve test otomasyonu)
- Test yönetimi (ör. Test tahmini, izleme ve risk yönetimi)
- Bilişim Teknolojileri ile ilgili eğitim (ör. Gereksinim mühendisliği, konfigürasyon yönetimi, proje yönetimi, sistem mühendisliği, yazılım geliştirme, geliştirme yaşam döngüsü modelleri)
- Kişilerarası beceriler (ör. İletişim, takım oluşturma)
- Alan uzmanlığı

Organizasyon ve proje testi eğitim ihtiyaçlarını uyumlu hale getirme (SP 1.2)

Test konusu için Uluslararası Yazılım Test Yeterlilik Belgesi (ISTQB) temel seviye ve ileri seviye eğitimleri şirket tarafından sağlanarak sertifikasyon verilme imkânı sunulabilir. Proje için proje paydaşları tarafından yapılan toplantılarla sistem mimarisi, işleyişi, neden yapıldığı, amacı, kapsamı gibi konuların eğitimi sağlanır.

Organizasyonel test eğitim planı oluşturulması (SP 1.3)

Test teknolojileri ile ilgili verilen eğitimlerin takibi sağlanır. Şirket işleyişinde test süreci ile ilgili durumlarda şirkete faydalı olabilecek eğitimlere katılacak kişiler belirlenerek plan oluşturulur.

Test eğitim kabiliyetinin oluşturulması (SP 1.4)

Test eğitim kabiliyetinin oluşturulması test organizasyonunda bulunan kişilerce düzenli bir hale getirilir. Test teknolojisinde yaşanan gelişmelerin takip edilmesi(test ile ilgili blogların takip edilmesi vb.) organizasyon içerisinde paylaşması ile eğitim ihtiyaçları belirlenir. Bu doğrultuda eğitim kabiliyeti oluşturulması bu uygulama ile sağlandı. Test eğitimin kurum içerisinde ya da dışarıdan alınması gerektiği belirlenir.

Test Eğitimi Sağlama (SG 2)

Test eğitimi verilmesi (SP 2.1)

Test eğitimleri belirlendikten sonra kimlerin eğitimlere katılması gerektiği belirlenir.

Test eğitim kayıtlarının oluşturulması (SP 2.2)

Test eğitimleri için katılacak çalışanların eğitim kayıtları oluşturulur.

Test Yaşam Döngüsü ve Entegrasyonu

Test yaşam döngüsü ve entegrasyon, kullanılabilir bir dizi organizasyonel test süreci varlığını (örneğin standart bir test yaşam döngüsü) ve çalışma ortamı standartlarını oluşturmak ve sürdürmek ve test yaşam döngüsünü geliştirme yaşam döngüsü ile entegre etmek ve senkronize etmek için tüm uygulamaları ele almaktadır. Entegre yaşam döngüsü bir projeye testin erken katılımını sağlar. Test yaşam döngüsü ve entegrasyonun amacı, tanımlanmış olana göre birden fazla test seviyesinde tutarlı bir test yaklaşımı tanımlamaktır.

Test Yaşam döngüsü ve Entegrasyonu Süreç Alanı için Özel Hedefler(SG) ve Özel Uygulamalar(SP)

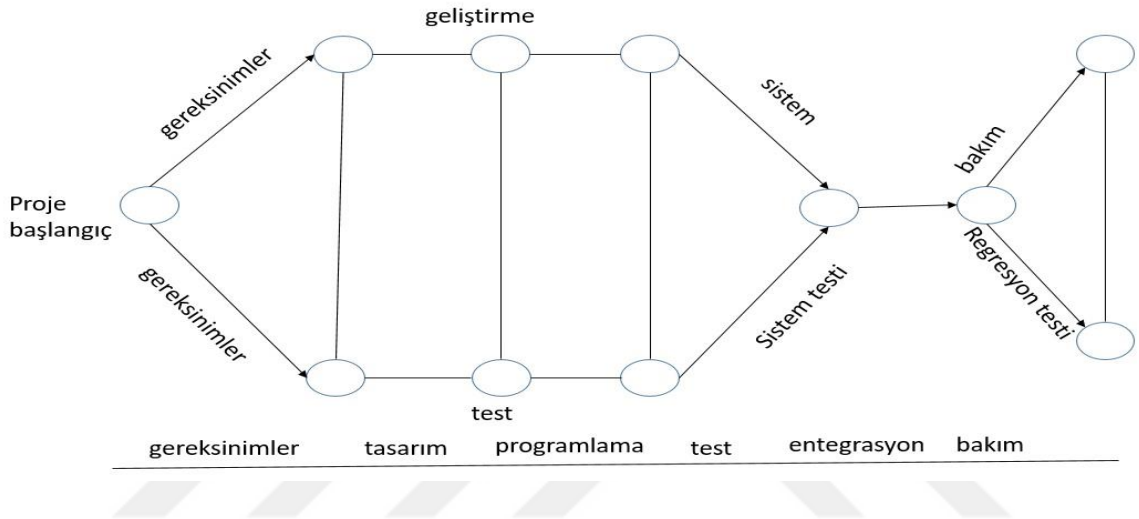
Örgütsel Test Süreci Varlıklarını Kurma (SG 1)

Standart test süreçleri kurmak (SP 1.1)

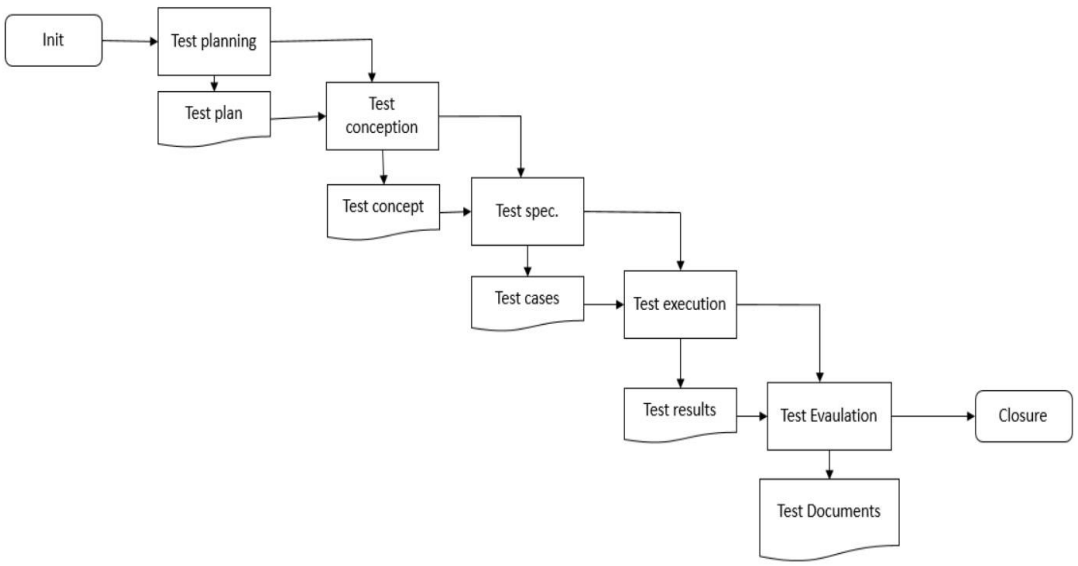
Seviye 2 deki süreç alanları ve uygulamaları kullanılarak standart bir test süreci kurulur.

Tüm test seviyelerine hitap eden test yaşam döngüsü model açıklamaları oluşturmak (SP 1.2)

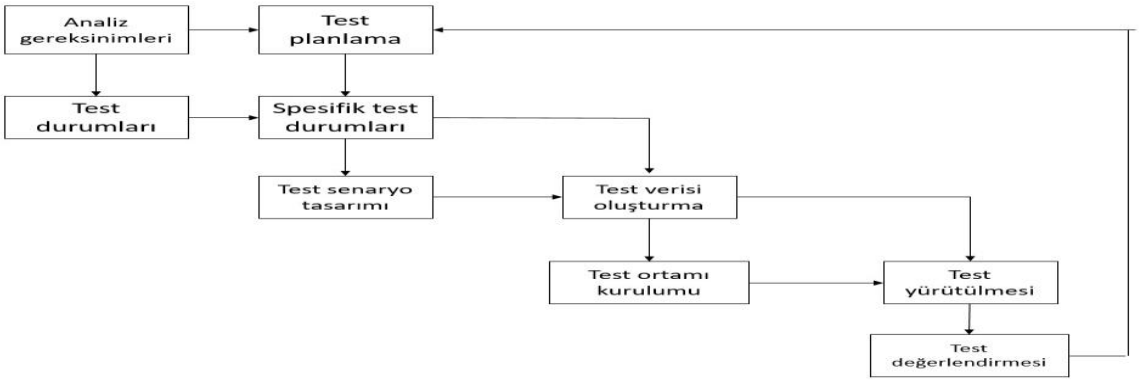
Test yaşam döngü modeli bu tez kapsamında Şekil 8.10' da test süreçlerinde yaşam döngü modeli, Şekil 8.11' de test süreci standart hale getirilmesi ve Şekil 8.12'de ise test seviyesi olarak sistem testi döngü modeli verilmiştir.



Şekil 8.10. Test süreçleri yaşam döngü modeli



Şekil 8.11. Standart bir test süreci modeli



Şekil 8.12. Sistem test döngüsü modeli

Uygunluk kriterlerini ve kurallarını oluşturmak (SP 1.3)

Uygunluk kriterleri ve yönergeleri:

- Kurumun standart test süreçleri ve organizasyonel test süreci varlıkları, özel olarak tanımlanmış test süreçlerini oluşturmak için nasıl kullanılır?
- Özel olarak tanımlanan süreçlerle karşılanması gereken zorunlu şartlar
- Kullanılabilecek seçenekler ve seçenekler arasından seçim yapma kriterleri
- Test süreci terzilerinin gerçekleştirilmesi ve belgelenmesinde izlenecek prosedürler

Kuruluşun test süreci veri tabanını kurma (SP 1.4)

Test süreci veri tabanı, test süreçlerinde ve sonuçta ortaya çıkan iş ürünlerinde veri toplamak ve mevcut hale getirmek için kurulmuştur. Test süreci ve iş ürün verileri:

- Test tahminleri ve gerçek veriler, örneğin, boyut, efor ve maliyet
- Kalite ölçütleri, ör. Öncelik seviyesi tarafından bulunan hataların sayısı
- Eş gözden geçirme değerlendirmesi kapsamı
- Test kapsamı
- Güvenilirlik önlemleri

Kuruluşun test süreci varlık kütüphanesi kurma (SP 1.5)

Kuruluşun test süreci varlık kitaplığında depolanacak öğeler:

- Test politikası ve test stratejisi
- Tanımlanmış test süreci açıklamaları
- Prosedürler (ör. Test tahmin prosedürü)
- Şablonlar
- En iyi uygulamalar test süreci varlıkları
- Tamamlanmış test planları
- Eğitim malzemeleri
- Proses yardımcıları (ör. Kontrol listeleri)
- Öğrenilen dersler (ör. Test değerlendirme raporları)

Çalışma ortamı standartlarını belirleme (SP 1.6)

Çalışma ortamı standartları:

- Çalışma ortamının çalışması, güvenliği için prosedürler
- Standart iş ortamı donanımı ve yazılımı
- Standart uygulama yazılımı

Test Yaşam Döngüsü Modellerini Geliştirme Modelleriyle Entegre Edilmesi (SG 2)

Entegre yaşam döngüsü modelleri oluşturulması (SP 2.1)

Yazılı geliştirme süreçleri bölümünde verilmiş olan modeller ve yukarıda verilmiş test modelleri entegre olacak şekilde oluşturulur. Test döngüsü kurma amaçları aşağıdaki gibidir:

- Denetlenebilir bir yazılım geliştirme döngüsü olması ve bu döngülerin yönetilebilir ve ölçülebilir olmasını sağlamak.

- Somut hale getirilen problemler ile daha fazla yazılımcının daha kolay işbirliği içerisinde olmasına katkıda bulunmak

- Her bir yazılım parçasını etraflıca ve zamanında test ederek, en az hata ile geliştirme döngüsünün sonraki aşamalarına geçilmesi

- Bir birimde ya da yeni yapılan geliştirmenin sistemdeki diğer bileşenleri nasıl etkilediğini zamanında görebilmek

- Kaliteli kod yazılabilmesini sağlamak

- Ortaya çıkan yazılım ürünün, müşteri isteklerini ve gereksinimleri karşılayıp karşılamadığını doğrulamak.

Entegre yaşam döngüsü modellerinin gözden geçirilmesi (SP 2.2)

Bu uygulamada önceki bölümlerde verilmiş yazılım geliştirme modelleri, paydaşlar ile gözden geçirilir.

Fonksiyonel Olmayan Test

Fonksiyonel olmayan test süreç alanı, fonksiyonel olmayan bir ürün risk değerlendirmesi gerçekleştirmeyi ve tanımlanan fonksiyonel olmayan riskleri temel alan bir test yaklaşımı tanımlamayı içerir. Fonksiyonel olmayan test süreci alanının amacı test planlama, test tasarımı ve yürütme sırasında fonksiyonel olmayan testlerin dâhil edilmesi için test süreci yeteneğini geliştirmektir. Tanımlanmış fonksiyonel olmayan ürün risklerine dayanan bir test yaklaşımı tanımlayarak, fonksiyonel olmayan test spesifikasyonları belirleyerek ve fonksiyonel olmayan testlere odaklanmış yapılandırılmış bir test yürütme sürecini yürüterek yapılır.

Fonksiyonel Olmayan Test Süreç Alanı için Özel Hedefler(SG) ve Özel Uygulamalar(SP)

İşlevsel Olmayan Ürün Riski Değerlendirmesi Yapılması (SG 1)

İşlevsel olmayan ürün risklerini tanımlama (SP 1.1)

Fonksiyonel olmayan ürün risklerini tanımlamada aşağıdaki teknikler kullanılmıştır.

- Beyin fırtınası
- Uzman görüşmeleri
- Kontrol listeleri

Ürün riskleri:

- Sisteme belirli bir kullanıcı sayısı ile giriş yaptığında sistemin çökmesi
- Sistemin farklı browserlarda kullanılabilirliğinin olmaması
- Sisteme giriş aşamasında güvenliğin işlememesi
- Sistemin bakımının yapılamaması
- Sistemde aynı işlemin belirli sayıdaki kullanıcılarla aynı anda yapıldığında sistemin kilitlenmesi
- Sistemin yavaşlığı

İşlevsel olmayan ürün risklerini analiz edilmesi (SP 1.2)

İşlevsel olmayan risk kategorileri:

- Kullanılabilirlik
- Güvenilirlik
- Verimlilik
- Bakım
- Taşınabilirlik

Çizelge 8.19. Fonksiyonel olmayan ürün riskleri analizi

Risk No	**Risk Tanımı**	Çıkma Olasılığı(1-5)	Etkisi(1-5)	Risk Değeri(1-25)
R1	Toplu işlemlerde yavaşlık ve kesinti problemleri	2	1	2
R2	Cihaz katalog yönetiminde veri girişinde format hatalarının olması	1	3	3
R3	Menüler arası geçişlerde yavaşlık	2	1	2
R4	Sayfa yüklemelerin gecikmesi	2	1	2
R5	Entegre olduğu sistem ile yaşanacak entegrasyon sorunları	4	5	20
R6	Ekranda görünür alanlardan etiket ve uyarı mesaj içeriklerinde yazım hataları	1	1	1
R7	Deploy sonrası sistemin mevcut yapısında yaşanacak sorunlar	1	5	5

Fonksiyonel Olmayan Bir Test Yaklaşımının Kurulması (SG 2)

Aşağıdaki uygulamalar ile gerçekleştirilmiştir.

Test edilecek özellikleri tanımlama (SP 2.1)

CKSY sistemi kapsamında performans, yük ve stres testleri gerçekleştirilmiştir. Bu kapsamda sisteme ait aşağıdaki kapsam dikkate alınır:

- Sistemin response-request zamanları
- Sistem için ideal bant genişliği
- Sistemin dar boğazları
- Sistemi yayınlayacak server için ideal donanım yapısı
- Sistem için ideal yük
- Sistemin kaldıracağı maksimum yük

Fonksiyonel olmayan test yaklaşımını tanımlama (SP 2.2)

Fonksiyonel olmayan testler kod, script test araçları ile gerçekleştirildiğinden “beyaz kutu test tekniği” yaklaşımı, performans ve kullanılabilirlik tarafından yazılım fonksiyonlarının

fonksiyonel olmayan gereksinimlere uygunluğu test edildiğinden “kara kutu test teknik” yaklaşımı tanımlanmıştır. Aşağıdaki testler gerçekleştirilmiştir.

- Performans testi: CKSY sistem uygulamasının planlanan performans kriterleri içerisinde çalışıp çalışmadığı test edilir. Belirli performans hedefleri koyulur. Web sitesi kaynak planlamasını (sunucu özellikleri, network, sunucu sayısı, bant genişliği) aynı zamanda veya belirlenen süre içerisinde (örneğin beş dakika) beklediği maksimum kullanıcı sayısına göre yapar. Performans testi ile CKSY sisteminin belirlenen performans kriterlerini (anlık iki bin kullanıcı, bir dakikada bir milyon istek (request), vb.) belirlenen kullanıcı sayısı kadar kullanıcının gerçekten girdiği gibi bir şekilde simüle eder ve beklenen maksimum kullanıcı geldiğinde sistemin nasıl çalışacağı ile ilgili bilgi edinilmiş olur.

- Yük testi: CKSY sistem uygulamasının belirlenmiş yük karşısında nasıl tepki verdiğini (cevaplarda gecikme süresi, kullanıcı deneyimi, vb.) gözlemleyen testtir.

- Stres testi: Olabilecek en fazla sayıdaki kullanıcı ile aşamalı olarak sisteme yüklenilmesi ile gerçekleştirilen testlerdir.

Fonksiyonel olmayan çıkış kriterlerini tanımlama (SP 2.3)

İşlevsel olmayan ürün kalitesi özelliklerine ilişkin çıkış kriterleri:

- Kullanılabilirlik için: kullanıcı memnuniyeti, işlevleri yerine getirmek için ortalama süre
- Verimlilik için: ortalama yanıt süresi, hafıza kullanımı
- Sürdürülebilirlik için: değiştirmek için ortalama çaba, belgelerin kullanılabilirliği

Fonksiyonel Olmayan Test Analizi ve Tasarımı Yapılması (SG 3)

Fonksiyonel olmayan test koşullarını tanımlanması ve önceliklendirilmesi (SP 3.1)

Performans testi kapsamı olarak aşağıdaki durumlar temel alınmıştır:

- Test stratejisinin uygun şekilde belirlenmesi
- Yük seviyelerinin sistemi doğru yorumlayacak şekilde belirlenmesi
- Performans test ortamının hazırlanması için test ortamın stabil, tutarlı olması gereklidir.
- Yük senaryolarının hazırlanması
- Testlerin uygulanması ve raporlanması

Chrome platformunda (Windows) komut satırı kullanabiliyor olmak gerekmektedir.

Fonksiyonel olmayan test durumlarını tanımlamak ve öncelik vermek (SP 3.2)

Test durumları örnek olarak aşağıdaki gibi oluşturulmuştur. Bu iki test durumu belirlenen test araçları ile gerçekleştirilmiştir. Test yürütülmesi aşamaları bir sonraki uygulama belirtilmiştir.

Çizelge 8.20. Fonksiyonel olmayan test durumları

Test Durum No	**Test Durum adı**	**Doğrulama noktası**	**Durum**
1	Aynı anda 250 eş zamanlı http isteği yapıldığında sistemin tüm istekleri karşılama durumu	Sistem 5 saniyede tüm istekleri karşılamalı	Passed
2	Aynı anda 500 eş zamanlı kullanıcı erişimin 5 kere tekrarlı olarak sağlanması	Hedef tamamlanma süresi 10 saniyeden az	Passed

Gerekli spesifik test verilerinin tanımlanması (SP 3.3)

Performans, stres ve yük testleri için sanal kullanıcılar oluşturulmuştur.

Fonksiyonel Olmayan Test Uygulaması Hazırlığı (SG 4)

Fonksiyonel olmayan test prosedürlerini geliştirmek ve önceliklendirmek (SP 4.1)

Performans, stres ve yük testleri SOAPUI ve Apache Jmeter toolları ile gerçekleştirilmiştir. Jmeter ve SoapUI test araçları için aşağıda bilgiler verilmiştir.

- *Jmeter:* Web tabanlı uygulamaların test edilmesi için tasarlanmış ve sistemde bulunan farklı fonksiyonları için test aktivitelerini gerçekleştirecek şekilde geliştirilmiş bir Apache projesidir. Web uygulamasını kullanan kullanıcıların sunuculara yaptıkları web istekleri, Jmeter aracıyla kullanıcı gibi davranıp bu kaynakları talep ediyormuş gibi simüle edilir. Simüle edilen kullanıcıların web tabanlı geliştirilen uygulamayı kullanışları, web uygulamasına yapılan istek girdileri farklılaştırılarak birden fazla kullanıcı aynı zamanda senaryoyu gerçekleştiriyormuş gibi tasarlanır ve istenen boyutta bir yük oluşturulabilir.

- *SoapUI:* Fonksiyonel test yapmak, web servisi simüle etmek, web servislerini kontrol etmek, herhangi bir web servisini çağırmak için kullanılır. Açık kaynak kodlu bir fonksiyonel test aracı olan SOAPUI ile hızlı bir şekilde performans testleri oluşturulur ve otomatik fonksiyonel testleri hazırlanabilir. SoapUI temel olarak bellek, CPU, hedef hizmet yanıt süresi ve diğer faktörlere bağlı olarak, LoadTesti donanımın yönetebileceği sayıda sanal kullanıcı (Thread) ile çalıştırılmasını sağlar. İstenen değer ayarlanır ve çalıştırılır.

Apache Jmeter Test Plan kısmında yük testi için gerekli adımlar:

- Sanal kullanıcıların oluşturulması
- Sanal kullanıcıların CKSY sistemi web sunucusuna yükleme yapılması
- HTTP sayfası için bilgiler girilmesi
- Sayfalara istek oluşturulması
- Login durumu için gerekli bilgilerin tanımlanması

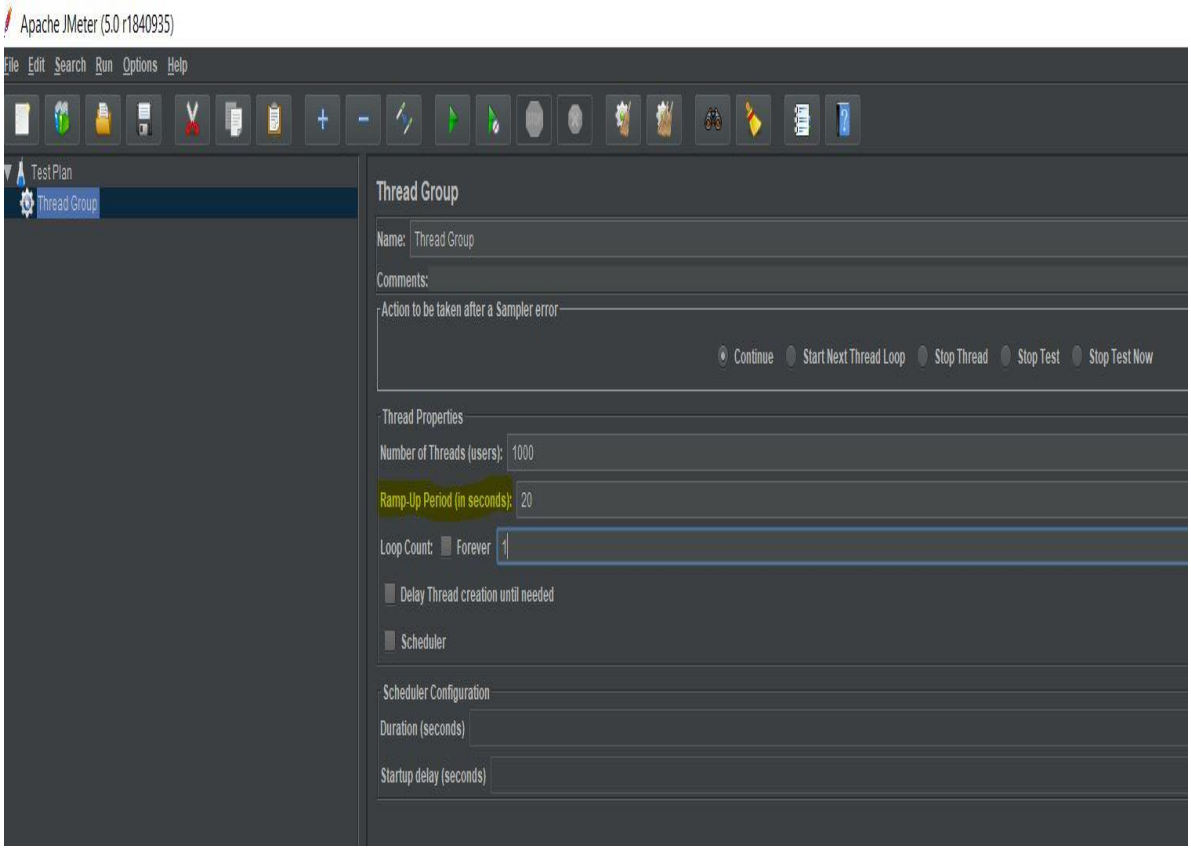
Özel test verileri oluşturulması (SP 4.2)

Test verileri tool üzerinden ilgili alanlar doldurularak oluşturulmuştur.

Fonksiyonel Olmayan Test Uygulaması Yürütülmesi (SG 5)

Fonksiyonel olmayan test senaryoları yürütme (SP 5.1)

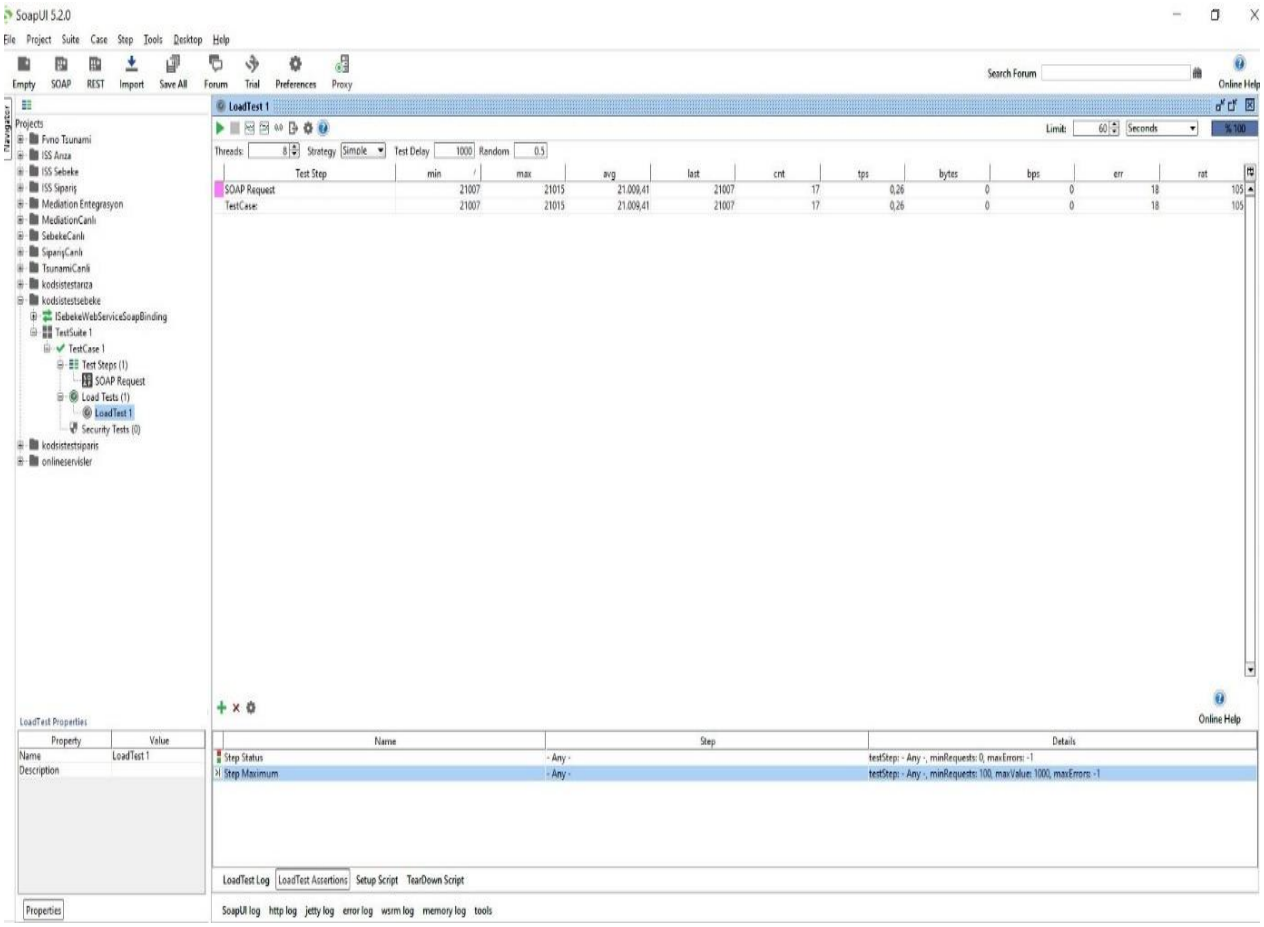
Bu uygulamada testler Apache Jmeter tollu kullanarak tanımlanan test durumları gerçekleştirilmiştir. Testte CKSY sistem uygulamasının 1000 kullanıcı için test edilmiştir. Sisteme ilk anda 1000 kullanıcı test aracı tarafından 20 saniye olarak verildi ve böylece ilk saniye bitiminde sistemde 50, ikinci saniye bitiminde 100 kullanıcı girmesi sağlandı ve 20 saniye bitiminde tüm kullanıcılar sisteme giriş yapmış olur.



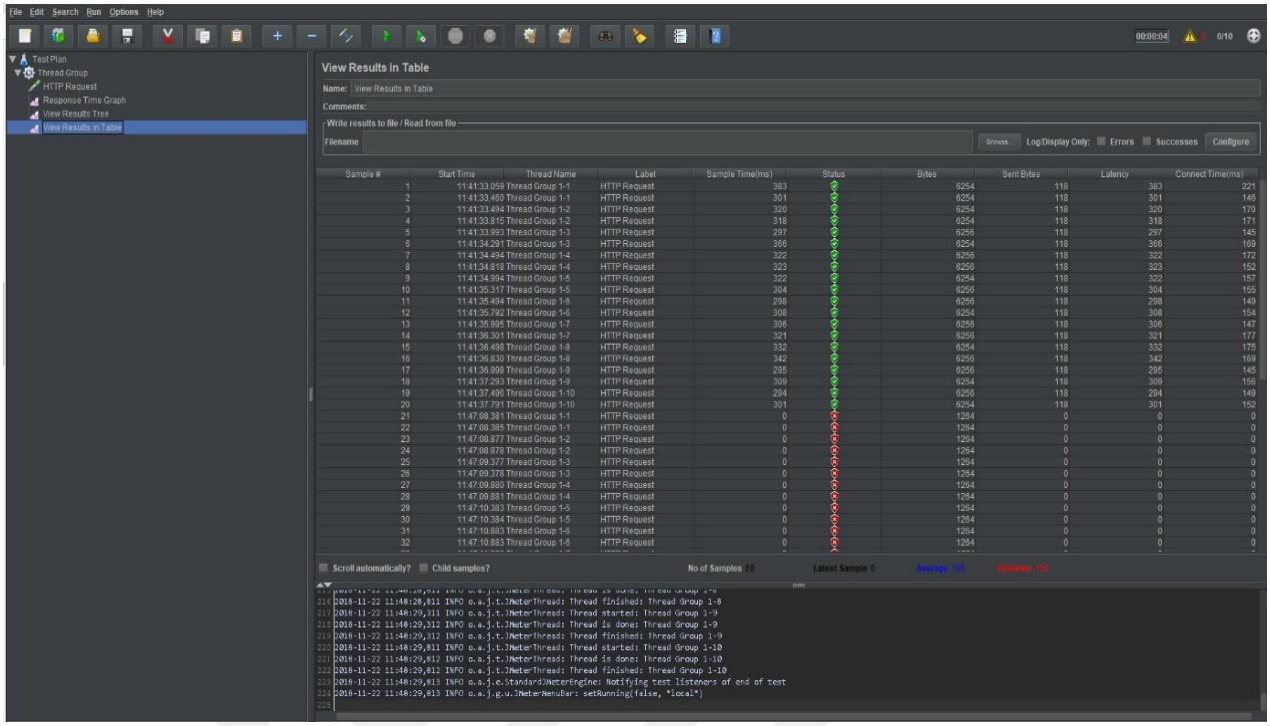
Şekil 8.13. Jmeter test toolu parametre giriş ekranı

Fonksiyonel olmayan test olaylarını rapor edilmesi (SP 5.2)

Fonksiyonel olmayan testler için yukarıda verilmiş raporlama için kullanılacak tablolar ile raporlama yapılabilir. Bu çalışmada fonksiyonel testler için kullanılan test araçlarının raporlamaları takip edilmiştir. Yük testi raporlaması için Resim 8.4' deki gibi SOAPUI uygulaması kullanılmış ve raporlamanın olduğu kısım aşağıda verilmiştir. Resim 8.5'te ise Apache Jmeter uygulamasında gerçekleştirilen ve test sonuçlarının olduğu ekran görüntüsü verilmiştir.



Şekil 8.14 SOAPUI test toolu load testi ekranı



Şekil 8.15. Apache Jmeter toolu test sonuçları görüntüsü ekranı

Eş Gözden Geçirme

Eş gözden geçirme süreci alanının amacı, iş ürünlerinin belirtilen gereksinimleri karşıladığını ve seçilen iş ürünlerindeki hataları erken ve etkin bir şekilde ortadan kaldırdığını doğrulamaktır. Gözden geçirmeler, değişikliklerin gerekli olduğu alanları ve hataları tanımlamak için meslektaşları tarafından iş ürünlerinin yöntemsel olarak incelenmesini içerir. İncelemeler, genellikle 2-7 kişi arasında, küçük bir mühendis grubuyla yapılır. İncelenecek olan iş ürünü, bir şartname, tasarım belgesi, kaynak kodu, test tasarımı, kullanım kılavuzu veya başka bir belge türü olabilir. Pratikte hakem grubunun seçildiği birçok yol vardır. Yorum yapanlar:

- Gözden Geçirme alanında uzmanlar (kalite güvence veya denetim)
- Aynı projeden insanlar
- Yazarı tarafından belirli bilgileri nedeniyle davet edilen kişiler
- İnsanlar, ör. Ürüne özel ilgi gösteren işletme temsilcileri

Bir örnekte yazar bir grup insanı bir dokümana ve onun düşünce sürecine yönlendirir, tüm çalışanlar dokümanı anlar ve yapılacak değişikliklerle ilgili bir fikir birliğine varır. Teknik bir incelemede grup bireysel bir hazırlık yapıldıktan sonra kullanılacak içerik ve teknik yaklaşımı tartışır. En resmi inceleme türü olan bir inceleme, bir belgenin her birey ve grup tarafından kaynaklar ve standartlar kullanarak ve öngörülen kuralları izleyerek hatalar için kontrol edildiği bir tekniktir.

Eş Gözden Geçirme Süreç Alanı için Özel Hedefler(SG) ve Özel Uygulamalar(SP)

Gözden Geçirme Yaklaşımının Kurulması (SG 1)

Gözden geçirme sürecinde statik test teknikleri yaklaşımı kurulmuştur.

İncelenecek iş ürünlerini tanımlanması (SP 1.1)

Gözden geçirme kapsamında ele alınan ürünler;

- Gereksinim tanımlama dokümanı
- Çözüm tasarım dokümanları
- Tanımlanan ürün riskleri
- Test plan dokumanı
- Hata raporları
- Kod gözden geçirmesi

Gözden geçirme kriterlerini tanımlanması (SP 1.2)

Giriş kriteri:

- Gereksinim dokümanı
- Kod
- Talep tanım dokümanı

- Tasarım dokümanı
- Test dokümanları

Çıkış kriterler ise;

- Gereksinim dokümanlarında maddelerin açık, anlaşılır ve kesin olması
- İş akış şemalarının olması
- Test durumları oluşturulacak düzeyde olması
- Tasarım dokümanlarında maddelerin teknik detaylarının açık olması(Ör. Tablo isimleri, etkilenen sistemler ve alanları
- Test dokümanlarının ise gereksinimleri kapsayacak şekilde oluşturulması
- Her gereksinim maddesine karşılık gelen test durumlarının bulunması

Gözden geçirme gerçekleştirilmesi (SG 2)

Gözden geçirme yürütülmesi (SP 2.1)

Gözden geçirme süreci sırasıyla Şekil 8.16'daki gibi; planlama, bilgilendirme, bireysel hazırlık, grup toplantısı, tekrar çalışma, izleme adımları izlenerek gerçekleştirilmiştir.

Şekil 8.16. Fagan'ın 6 adımlı gözden geçirme süreci [17].

Test uzmanlarının temel test dokümanlarını gözden geçirmesi (SP 2.2)

Yukarıdaki süreç izlenerek grup gözden geçirme yöntemlerinden inceleme yöntemi ile gözden geçirme yapılmıştır.

İnceleme(Inspection): Planlı, dokümanlar ve araçlar yardımı ile yapılan bu yöntemde saptanan hatalar raporlanır ve düzeltilmesi için yazılımlara iletilir. Böylece inceleyiciler ile

kodu yazan ortak kanıya vardıklarında süreç tamamlanır ve onaylanan kod, doküman bir sonraki sürece geçer. Gözden geçirme sürecinde roller:

- Lider: Gözden geçirme toplantılarını yönetir.
- Yazar: Geliştirici ya da proje ile ilgili herhangi bir doküman sahibi
- İnceleyici: Geliştirme yapan ve doküman inceleyen uzmanlaşmış kişiler
- Yönetici: Planlamaları yapan, yöneten ve iletişim kuran kişi
- Yazıcı: Toplantı sırasındaki tüm notları, bulguları kaydeder.

Çizelge 8.21. Eş gözden geçirme kontrol maddeler formu

		Kontrol Sonucu	
Madde No	**Madde**	**Evet**	**Hayır**
1	Gereksinim, sunulacak işlevselliği doğru olarak tanımlıyor mu?		
2	Doğrulaması yapılmakta olan gereksinim, çıkartıldığında değişen bir şey olmuyor mu ya da bir işlevsellik yerine getirilemiyor mu?		
3	Gereksinim, geliştirilecek sistem için zorunlu mu?		
4	Gereksinimler, subjektif ifadeler içeriyor mu? (Örnek:"güzel", "iyi", "hızlı" gibi ifadeler ölçümlemesi yapılamayacak subjektif ifadelerdir.)		
5	Projenin bütçe, zaman ve insan kaynağı ile gerçekleştirilemeyecek gereksinim var mı?		
6	Anlaşılması güç ifadeler kullanılmış mı?		
7	Bütünlüğü bozan veya birbiriyle çelişen gereksinim var mı?		
8	İhtimal içeren kelimeler kullanılmış mı?(olabilir, belki)		
9	"çoğunlukla", "bazen", "ara sıra", "genellikle" gibi ifadeler kullanılmış mı?		
10	Kullanıcı dostu", "çok yönlü", "esnek", "yaklaşık", "tahminen", "etkili", "verimli" kişiden kişiye değişen (göreceli) kelimeler kullanılmış mı? (bu kelimeler gereksinimin "nasıl" gerçekleştirileceği konusunda belirsizlik yaratmaktadır.)		
11	Gereksinimlerde ve veya bağlaçları kullanılmış mı?(Kullanıldı ise bunlar 2 ayrı gereksinim olarak yazılmalı)		
12	Gereksinim, Gereksiz, uzun cümlelerden oluşan ve konuyu dağıtan ifadelere sahip mi?		
13	Terim kullanılmış mı? (Mümkün olduğunca terimler kullanılmamalıdır. Kullanılması gereken terimler, terim sözlüğü gibi ortak bir yerde detaylarıyla açıklanarak, gereksinimler içerisinde adreslenmelidir.)		

Gözden geçirme verilerinin analiz edilmesi (SP 2.3)

Araçlarla yapılan statik kod analizleri ile hatalar önce bulunarak, karmaşık kodlar bulunarak, dinamik testlerde bulunması zor olan hatalar ve tutarsızlıklar belirlenir. Statik analiz yapıldıktan sonra saptanan hatalar yazılımın kod yapısıyla ilgilidir ve bulunan hatalar çözülmediği durumda yazılıma maliyeti artacaktır. Kullanılan yazılım diline göre değişebilen bu tür araçları kullanabilmek için yazılım dilini iyi bilmek gerekir ve bu nedenle araçlar kod yazanlar tarafından kullanılır. Bu araçlar;

- Derleyici(Complier): Bu araç ile yazılan kodlar derlenir ve çalıştırılır.
- Cyclomatic Complexity*:* Kodun karmaşıklık seviyesini gösteren araçlardır.
- Akış Şeması(Control Flow): Kodun çalıştırılırken nasıl çalışacağını gösteren yol akış şemasını içerir.
- Veri Akışı(Data Flow): Bu araçta veriler ile ilgili oluşturma, kullanma, silme adımlarını anlatan akış şemaları vardır.
- Statik Analiz(Static Analysis): Yazılımı oluşturan parçaların çalıştırılmadan analiz edilmesidir.

8.2.2. Seviye 3 düzeyinin değerlendirilmesi ve olgunluk seviye belirlenmesi

Seviye 3 için tanımlı olan süreçler, uygulamalar ile tamamlanmıştır. Aşağıda seviye 3 için yapılan ve elde edilenler özetlenmiştir.

- TMMi Seviye 3 ile test artık kodlamayı takip eden bir faz olmaktan çıkarılmıştır. Yazılım geliştirme yaşam döngüsüne entegre edilmesi sağlanmıştır.
- Test planlaması erken bir proje aşamasında yapılmıştır. Örneğin, gereksinimler fazı sırasında ve bir ana test planında belgelendirilmiştir. Test planının yapılması, 2nci olgunluk seviyesinde elde edilen test planlama becerilerini geliştirmiştir. Organizasyonun, olgunluk seviyesi 3'ün temeli olan standart test süreçleri seti, zaman içinde kurulur ve geliştirilir.

- Bir test organizasyonu ve özel bir test eğitim programı bulunması sağlanmış ve test bir meslek olarak algılanmaya başlamıştır.

- Test süreci iyileştirme, test kuruluşunun kabul ettiği uygulamaların bir parçası olarak kurumsallaştırılmıştır.

- Gözden geçirmelerin gerçekleştirilmesi ve bu seviyede gereken önemi ile birlikte yaşam döngüsü boyunca gerçekleştirilmiştir.
- Test uzmanları, gereksinim özelliklerinin gözden geçirilmesinde yer almaya başlamıştır.

- TMMi 2 seviyesindeki test tasarımları temel olarak fonksiyon testlerine odaklanırken, test tasarımları ve test teknikleri, fonksiyonel olmayan testler de dâhil olmak üzere 3. seviyede genişletilmiştir. Böylece test kuruluş içerisinde ve uygulanan sistem de bulunan paydaşlar tarafından benimsenmiş ve test yazılım geliştirme yaşama döngüsüne entegre edilmesi sağlanmıştır.

- Mevcut durumda karşılaşılan problemlerde belirgin azalmaların olduğu gözlemlenmiştir. Canlıdan gelen hata sayılarında büyük ölçüde düşüş olduğu ve dolayısıyla üst yönetim baskısının giderildiği, müşteri memnuniyetinden geri bildirimlerin olumlu olmaya başladığı ve yazılım geliştirme süreçlerindeki etkili, sürdürülebilir ve verimli bir süreç işletilmeye başlatıldığı gözlemlenmiştir.

TMMi modeli seviye 3 için verilmiş olan süreç alanlarına özel hedefler ve özel uygulamalar gerçekleştirildikten sonra tez kapsamında önerilen test olgunluk seviye ölçümü gerçekleştirildi. Sonuçlar aşağıdaki tabloda verilmiştir. Olgunluk seviyesi ölçümü için paydaşlar tarafından sorular cevaplandırılmış ve aşağıdaki tablodaki gibi puanlamalar yapılmıştır.

Çizelge 8.22. TMMi 3. Seviye değerlendirme için verilen ortalama puanlar ve başarı puanı

TMMi 3.Seviye	Verilen Puanların Ort.
1.Tanımlanmış test organizasyonu, test rolleri ve sorumlulukları bulunuyor.	4,57
2.Test süreç iyileştirme planlamaları bulunuyor.	4,71
3.Yazılım test alanında eğitim imkânları sağlanıyor ve eğitim planları uzmanlık seviyelerine göre oluşturuluyor.	4,42
4.Test seviyelerine hitap eden tanımlanmış test yaşam döngüleri bulunuyor.	3,85
5.Tanımlanmış test süreç veri tabanı ve kütüphanesi bulunuyor.	3,28
6.Şirketinizde yazılım geliştirme yaşam döngülerine entegre edilmiş test süreci bulunuyor.	4,14
7.Fonksiyonel olmayan testler için resmi test yaklaşımları (test edilecek öğeler, giriş ve çıkış kriterleri vb.) bulunuyor.	4,42
8.Fonksiyonel olmayan testler (Stres, Yük, Güvenlik, Performans, Kullanılabilirlik vb.) gerçekleştiriliyor.	4,42
9.Kurulmuş bir gözden geçirme süreci yaklaşımı bulunuyor.	4,28
10.Statik test teknikleri olarak gözden geçirme süreçleri işletiliyor ve statik test analizleri yapılıyor.	4,42
ORTALAMA	**4,251**
BAŞARI	**85,02**

Proje kapsamında çalışanlar tarafından sorulara verilen puanların ortalama değerleri ve Seviye 3 için başarı yüzdesi %85,02 olarak çıktığı tabloda görülmektedir. Bu sonuca göre CKSY projesi üzerinde test olgunluk seviyesi Seviye 3 olarak belirlenmiştir.

9.SONUÇ VE ÖNERİLER

Bu çalışmada yazılım geliştirme süreçleri ve yazılım test süreçleri derlenmiş, yazılım test süreçleri iyileştirme olgunluk modellerine yer verilmiştir. Daha sonra CMMI modeli ve Test Olgunluk Modeli Entegrasyonu (TMMi) özetlenmiştir.

Test olgunluk seviyesi belirleme ve değerlendirme için bir yöntem verilmiş ve uygulama ile geçerliliği sınanmaya çalışılmıştır. Test olgunluk seviyesi yükseltmek için TMMi modeli uygulamalı olarak bir sistem üzerinde detaylı olarak ele alınmıştır. TMMi modelinin yapısı, amacı, kapsamı, bileşenleri ve olgunluk seviye modelleri ile birlikte süreç alanları ve bu süreç alanlarına uygulanan özel uygulamalar TMMi vakfının hazırlamış olduğu doküman referans alınarak bu çalışma da ayrıntılı olarak verilmiştir.

Sonuç olarak TMMi modelinin Seviye 2 düzeyi, 5 adet süreç alanlarında belirtilen 17 adet özel hedeflere 63 adet özel uygulamalar sırasıyla gerçekleştirilerek sağlandı. Bu uygulamalar da kanıtlar ve dokümanlar oluşturulmuştur. Sonuç olarak Seviye 2 deki süreç alanları bu uygulamalar ile sağlanarak proje ve kuruluşun test olgunluk seviyesi 2 ye yani yönetilen düzeye çıkarılmıştır. Seviye 2 değerlendirme ve belirleme için tez çalışması kapsamında verilen yöntemde seviye 2 değerlendirme ölçümü kullanılarak geçerliliği sınanmıştır. Sonrasında olgunluk seviyesi 2'den olgunluk seviyesi 3' e geçiş süreci gerçekleştirilmiştir. Seviye 3 düzeyi, belirlenen 5 adet süreç alanlarında bulunan 14 adet özel hedeflere 39 özel uygulamalar gerçekleştirilerek sağlanmıştır. Bu uygulamalar da kanıtlar ve dokümanlar oluşturulmuştur. Sonuç olarak TMMi Seviye 3 ile test artık kodlamayı takip eden bir faz olmaktan çıkarılarak ve yazılım geliştirme yaşam döngüsüne entegre edilmesi sağlanmıştır.

Sonuç olarak verilen olgunluk belirleme yöntemi ile danışmanlık hizmeti alınmadan test olgunluk seviyesi belirlenmiştir. Olgunluk seviye yükseltilmesi için TMMi modeli uygulanarak projeyi geliştiren organizasyonda bütün ekipçe benimsenen, sürdürülebilen ve etkili bir test süreci sağlanmıştır. Yeni geliştirmeler, değişiklikler ya da hata düzeltmelerinde test aktiviteleri kişilerin yetkinlikleri ve inisiyatiflerine bağlı şekilde ilerletilmesinin önüne geçilmiştir. Canlıdan gelecek hataları azaltılması, tekrarlı eforların önüne geçilmesi, yönetim baskısı giderilmesi ve müşteri memnuniyeti sağlanması bu uygulama sonucunda görülmüştür. Bu tez çalışmasında yazılım sektöründe bulunan şirketlere test süreç

iyileştirmelerinin önemi ve bunu TMMi modeli ile sağlayabilecekleri ve firmaların yüksek maliyetlere girerek yaptıracakları değerlendirme ve iyileştirme çalışmalarına başlamadan kendi süreçlerinin değerlendirmelerini yapabilecekleri tez kapsamında verilen yöntem ile gösterilmiştir. Bu tez çalışması kullanılarak firmalar, olgunluk seviyesi belirleme ve değerlendirme çalışmalarını ve test süreç iyileştirme çalışmaları yapabilirler. Tez çalışmasının devamı olarak Seviye 4 ve Seviye 5 olgunluk düzeyine ulaşılması hedeflenmektedir.

KAYNAKLAR

1. Alparslan, S.,A. (2017). *CMMI ile Yazılım Süreçlerinin İyileştirilmesi ve Yazılım Şirketlerinin CMMI 3 Seviyesine göre Değerlendirilmesi* (Yüksek Lisans Tezi, Alanya Alaaddin Keykubat Üniversitesi Fen Bilimleri Enstitüsü 2017).

2. Gül, Z. (2006). *Yazılım Geliştirme Sürecinin İyileştirilmesi ve Türkiye Uygulamaları* (Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü 2006).

3. Garousi, V. and Felderer, M. (2017). Software test maturity assesment and test process improvement: A mutivocal literature review. *Information and Software Technology,* 85(1), 2-18.

4. Sudarsanam, S. K. (2013). Software Test Process Assesment Methodology. *Mühendislik, Bilgisayar ve Uygulamalı Bilimler Dergisi,* 2(6), 52-56.

5. İnternet: Barış Sarıalioğlu, Berk Dülger. Test Olgunluk Seviyesi Modeli. URL: http://ceur-ws.org/Vol-1483/9_Bildiri.pdf, Son Erişim Tarihi: 08.04.2019.

6. Arkan, S. (2016). *Assessing The Maturity Of Software Testing Services: A Model and Its Industrial Evaluation* (Yüksek Lisans Tezi, Çankaya Üniversitesi Fen Bilimleri Enstitüsü 2016).

7. Serdaroğlu, D. (2015). *Yazılım Test Süreci ve TMMi Modelinde Prisma Yaklaşımı Uygulaması* (Yüksek Lisans Tezi, Başkent Üniversitesi Fen Bilimleri Enstitüsü 2015).

8. Hancı, A. K. (2017). *Yazılım Test Tasarım Tekniklerinin Matematiksel Model Yaklaşımı ile Analizi* (Yüksek Lisans Tezi, İstanbul Üniversitesi Fen Bilimleri Enstitüsü 2017).

9. Kahveci, G. (2017). *Bankacılık Alanında Kişisel Yazılım Test Eforunu Tahmin Etmek için Proxy Tabanlı bir Metot ve Vaka Çalışması* (Yüksek Lisans Tezi, Hacettepe Üniversitesi Fen Bilimleri Enstitüsü 2017).

10. Baran, A. , Akar, F. and Aslay, F. (2016,Temmuz). Çevik Yazılım Geliştirme Yöntemlerinde Etkili Test Araçları. ISTEC Uluslararası Bilim ve Teknoloji Konferansı, Avusturya.

11. Kayacan, A. (2017). *Şelale Modeli ve Çevik Yöntemlerin Yazılım Proje Yönetimi Bakış Açısıyla İncelenmesi ve Karşılaştırılması* (Yüksek Lisans Tezi**,** Gazi Üniversitesi Bilişim Enstitüsü 2017).

12. İnternet: Yagup Macit, Eray Tüzün. Uygulama Yaşam Döngüsü Yönetimi Karşılaştırmalı Süreç İncelemesi. URL: http://ceur-ws.org/Vol-1483/12_Bildiri.pdf , Son Erişim Tarihi: 09.04.2019.

13. Gencer, C. and Kayacan, A. (2017). Yazılım Proje Yönetimi: Şelale Modeli ve Çevik Yöntemlerin Karşılaştırılması. *Bilişim Teknolojileri Dergisi***,** 10(3),337-339.

14. Yücalar, F. ve Borandağ, E. (2019). Yazılım Projelerinde Kalitenin Arttırılması:TMMi. *AURUM Mühendislik Sistemleri ve Mimarlık Dergisi*, 2(3),77-81.

15. Bose, S.C. ve Bose,G. (2016). Transforming organizations to achieve TMMi certification. Thirty-Fourth Annual Pacific Northwest Software Quality Conference, Portland.Oregon.USA.

16. Araujo,A.F.,Rodrigues,C.L.,Vincenzi,A.M.R.,Camilo,C.G. and Silva, A.F. (2013). A Framework for Maturity Assessment in Software Testing for Small and Medium-Sized Enterprises. International Conference on Software Engineering Research and Practice.

17. Gürbüz, A.(2010). *Yazılım Test Mühendisliği* (Birinci Baskı). Türkiye: Papatya Yayıncılık Eğitim, 14-28, 39-58,63-72,74-78,81-88.

18. Ocak, Ş. ve Yıldıztekin, M. (2011). Güvenli Yazılım Geliştirme Modellerinin Karşılaştırılması. Elektrik Elektronik ve Bilgisayar Sempozyumu, Elazığ.

19. TMMi Foundation, TMMI Framework and Levels, 2009, www.tmmifoundation.org

20. TMMi Foundation, TMMi Assessment Method Application Requirements (TAMAR) Version 2, 2009, www.tmmifoundation.org

21. ISTQB Foundation Syllabus ,International Software Testing Qualification Board,2011. https://www.istqb.org/downloads/syllabi/foundation-level-syllabus.html

22. Sarıdoğan, M.A (2011). *Yazılım Mühendisliği* (Üçüncü Baskı).Türkiye: Papatya Bilim.

23. KAN, S.H. (1995). *Metrics and Models in Software Quality Engineering* (İkinci Baskı). USA: Addison Wesley Publishing.

24. İnternet: Ayfer Başar. Test Süreçlerinin Olgunluk Seviyesi Modeli ile İyileştirilmesi: Scrum ile Yazılım Geliştiren Bir İşletmede Uygulama. URL: https://docplayer.biz.tr/6111354-Test-sureclerinin -olgunluk-seviyesi-modeli-ile-iyilestirilmesi-scrum-ile-yazilim-gelistiren-bir-isletmede-uygulama.html. Son Erişim Tarihi: 09.04.2019.

ÖZGEÇMİŞ

Kişisel Bilgiler

Adı- Soyadı: Gökhan ŞİT

Uyruğu: Türkiye Cumhuriyeti

Doğum Tarihi ve yeri: 02.03.1991-Erzincan

Medeni Hali: Bekâr

e-posta: gkhnsit@gmail.com.tr

Eğitim Derecesi	**Okul/Program**	**Mezuniyet yılı**
Lisans	Atatürk Üniversitesi	2013
İş Deneyimi/Yıl	**Çalıştığı Yer**	**Görevi**
2015-2018	Ericcson	Test Uzmanı
2018-	Türksat	Test Uzmanı

Yabancı Dil

İngilizce

Sertifikalar

Bölgesel Kalkınma Çalıştayı-Atatürk Üniversitesi-2011

Genç Girişimcilik Programı- Atatürk Üniversitesi Endüstri Mühendisleri Kulübü-2012

Kalite Çemberleri Katılım Belgesi-Atatürk Üniversitesi Mühendislik Fakültesi-2012

ISTQB-Turkish Testing Board-2014

IIBA CBAP – CCBA Uluslararası-2018