T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI
ENDÜSTRİ MÜHENDİSLİĞİ BİLİM DALI

PROJE YÖNETİM SÜRECİNİN İYİLEŞTİRİLMESİ: BİLİŞİM SEKTÖRÜNDE BİR UYGULAMA

(Yüksek Lisans Tezi)

Tezi Hazırlayan:

Burcu TANRIKUT ÇERKEZOĞLU

İSTANBUL, 2017

T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
ENDÜSTRİ MÜHENDİSLİĞİ ANABİLİM DALI
ENDÜSTRİ MÜHENDİSLİĞİ BİLİM DALI

PROJE YÖNETİM SÜRECİNİN İYİLEŞTİRİLMESİ: BİLİŞİM SEKTÖRÜNDE BİR UYGULAMA

(Yüksek Lisans Tezi)

Tezi Hazırlayan:

Burcu TANRIKUT ÇERKEZOĞLU

Öğrenci No:

140792001

Danışman:

Prof. Dr. Semra BİRGÜN

İSTANBUL, 2017

YEMİN METNİ

Yüksek lisans tezi olarak sunduğum "Proje Yönetim Sürecinin İyileştirilmesi: Bilişim Sektöründe Bir Uygulama" başlıklı bu çalışmanın, bilimsel ahlak ve geleneklere uygun şekilde tarafımdan yazıldığını, yararlandığım eserlerin tamamının kaynaklarda gösterildiğini ve çalışmamın içinde kullanıldıkları her yerde bunlara atıf yapıldığını belirtir ve bunu onurumla doğrularım. 11.05.2017

Burcu TANRIKUT ÇERKEZOĞLU

T.C.
BEYKENT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

YÜKSEK LİSANS TEZ SAVUNMA SINAVI SONUÇ TUTANAĞI

Beykent Üniversitesi
Fen Bilimleri Enstitüsü Müdürlüğü'ne,

Aşağıda tez adı belirtilen yüksek lisans öğrencisi 140792001 no'lu Burcu Tanrıkut Gerezoğlu'nun tarihinde yapılan tez savunma sınavı[1] sonucunda 60 dakika süreyle sunduğu ve savunduğu tezi hakkında[2] oybirliği / ~~oyçokluğu ile~~, kabul kararı verilmiştir.

Bilgilerinize saygılarımızla arz ederiz.

Anabilim Dalı : Endüstri Mühendisliği
Programı : Endüstri Mühendisliği
Tez Başlığı[3] : Proje Yönetim Sürecinin İyileştirilmesi : Bilişim Sektöründe Bir Uygulama

---

Tez Sınav Jürisi	Öğretim Üyesi	İmza
Danışman	: Prof. Dr. Senna Birgün	
Üye	: Y. Doç. Dr. F. Serap Onursal	
Üye	: Y. Doç. Dr. A. Yekta Kayman	

[1] Jüri üyeleri söz konusu tezin kendilerine teslim edildiği tarihten itibaren en geç bir ay içinde toplanarak öğrenciyi tez savunma sınavına alır. Belirlenen günde yapılamayan jüri toplantısı, katılanların hazırladığı bir tutanakla enstitü yönetimine bildirilir. Bu durumda jüri en geç onbeş gün içinde toplanarak adayı tez savunma sınavına alır. Tez savunma sınav süresi en az 45 dakikadır. Yüksek lisans tez savunma sınavı, tez çalışmasının sunulması ve bunu izleyen soru-yanıt bölümlerinden oluşur ve dinleyiciye açıktır. (Beykent Lisansüstü eğitim ve Öğretim Yönetmeliği-Madde30-3)

[2] Tez sınavının tamamlanmasından sonra jüri, tez hakkında "kabul", "düzeltme" veya "red" kararı verir. Jüri başkanı, jüri üyelerince imzalanmış sınav tutanağını, tez sınavını izleyen üç gün içinde ilgili enstitü yönetimine teslim eder. Tezi hakkında düzeltme kararı verilen öğrenci en geç üç ay içinde gerekli düzeltmeleri yaparak ve yönetmelikte belirtilen usullere uygun olarak tezini aynı jüri önünde yeniden savunur. (Beykent Lisansüstü eğitim ve Öğretim Yönetmeliği-Madde30-4)

[3] İleride doğabilecek aksaklıkların engellenmesi için tezin başlığının yazılması gerekmektedir.

TEŞEKKÜR

Yüksek Lisans tez çalışmamda planlanmasında, araştırılmasında, yürütülmesinde ve oluşumunda ilgi ve desteğini esirgemeyen, engin bilgi ve tecrübelerinden yararlandığım, yönlendirme ve bilgilendirmeleriyle çalışmamı bilimsel temeller ışığında şekillendiren sayın hocam Prof. Dr. Semra BİRGÜN'e sonsuz teşekkürlerimi sunarım. Ayrıca çalışma süresince tüm zorlukları benimle göğüsleyen ve hayatımın her evresinde bana destek olan eşim Metin ÇERKEZOĞLU'na, annem Aysel TANRIKUT'a, babam Nuri Bora TANRIKUT'a ve kardeşim Cem TANRIKUT'a teşekkürlerimi bir borç bilirim.

İstanbul, 2017

Burcu TANRIKUT ÇERKEZOĞLU

PROJE YÖNETİM SÜRECİNİN İYİLEŞTİRİLMESİ: BİLİŞİM SEKTÖRÜNDE BİR UYGULAMA

Tezi Hazırlayan: Burcu TANRIKUT ÇERKEZOĞLU

ÖZET

Günümüzde yazılım sistemleri üretim, finans, sigorta, sağlık, ulaşım, finans ve daha birçok farklı alanda karşımıza çıkmaktadır. Firmaların çoğunda gerekli yazılımlar olmadan organizasyonun yaşamını devam ettirmesi mümkün gözükmemektedir. Organizasyonların artan rekabet ve gelişen teknoloji yüzünden yazılım sistemlerine olan ihtiyaçları her geçen gün artmaktadır. Değişen teknoloji ve müşterilerin talepleri, yazılım sistemlerini gün geçtikçe çok daha karmaşık hale getirmektedir.

Araştırmalara göre yazılım projelerinin büyük bir çoğunluğunun hatalı, güvenilirliği az, müşteri istek ve ihtiyaçlarına çözüm geliştiremeyen ve zamanında teslim edilmeyen yazılımları ürettiği gözlemlenmiştir. Temel sorunun yazılım geliştirme sürecindeki eksikliklerden kaynaklandığı ortaya konulmuştur.

Bu çalışmada, yazılım geliştirme süreci temel yönleriyle incelenmeye çalışılmıştır. Yalın teknikler ve Değer akış haritalama yöntemiyle birlikte yazılım geliştirme metodolojileri arasından seçim yapılması sağlanarak genel teorik bilgilere değinilmiştir. Bilişim sektöründe yalın bir uygulama örneği verilerek, değer akış haritalama yöntemi kullanılmıştır. Değer akış haritasının analizi ile hazırlık süreleri ve değer katmayan faaliyetler tespit edilmiştir. Oluşturulan gelecek durum akış haritası ile yalın tekniklerden faydalanılarak yazılım geliştirme süreci iyileştirilip, proje başarı durumuna göre analiz edilerek uygulama değerlendirilmiştir.

Anahtar Kelimeler: Proje Yönetimi, Süreç İyileştirme, Yazılım Geliştirme, Değer Akış Haritalama

IMPROVEMENT OF THE PROJECT MANAGEMENT PROCESS: AN APPLICATION IN SOFTWARE INDUSTRY

Presented by: Burcu TANRIKUT ÇERKEZOĞLU

ABSTRACT

Nowadays, software systems are emerging in manufacturing, finance, insurance, healthcare, transportation, finance and many other areas. In most companies, it is not possible to maintain organizational activities without software. Organizations are increasingly in need of software systems due to increased competition and improved technology. The change of technology and demans of customers make more complex day after day the software systems.

According to the researches observed the most of software projects are erroneous, less realiability, unable to develop solutions for customers needs and the software couldn't be delivered on time. The main problem is the lack of software development process.

In this study, the software development process examined in terms of basic aspects. Lean techniques and value stream mapping method, as well as software development methodologies, have been selected to address the general theoretical knowledge. An application example is given in the information sector and value stream mapping method is used. An analysis of the value stream chart has identified the preparation times and activities that do not add value. The software development process was improved by using the generated future state stream map and lean techniques and the project was analyzed according to the success status and this application was evaluated.

Keywords: Project Management, Process Improvement, Software Development, Value Stream Mapping

İÇİNDEKİLER

Sayfa No.

TABLOLAR LİSTESİ

Sayfa No.

ŞEKİLLER LİSTESİ

Sayfa No.

KISALTMALAR

ABD : Amerika Birleşik Devletleri

AHP : Analytic Hierarchy Process (Analitik Hiyerarşi Süreci)

A.Ş. : Anonim Şirketi

BT : Bilgi Teknolojileri

DAH : Değer Akış Haritalama

DAY : Değer Akış Yönetimi

EPE : Every Part Every (Üretim Parti Büyüklüğü)

FDD : Feature-Driven Development (Özellik Odaklı Geliştirme)

GDH : Gelecek Durum Haritası

PROD : Production (Canlı Ortam)

RUP : Rational Unified Process (Birleşik Rasyonel İşlem)

TDK : Türk Dil Kurumu

UAT : User Acceptance Testing (Kullanıcı Kabul Testi)

XP : Extreme Programming (Aşırı Programlama)

YBS : Yönetim Bilişim Sistemleri

1.GİRİŞ

Her alanda küreselleşmenin ve rekabetin giderek arttığı dünyamızda müşteri memnuniyeti sağlamak ve sadık müşterilere sahip olmak oldukça önemlidir. Müşterilere sunulan her mal veya hizmet belli bir süreç aşamasından geçerek önlerine sunulmaktadır. Bu süreç çıktısının müşterinin istek ve beklentilerine uygun olması ve bu aşamaları en az maliyetle oluşturabilmek için sürecin her aşaması incelenmelidir.

Bilgi çağı ile endüstriyel çağın rekabetinin temel varsayımlarının birçoğu geçerliliğini yitirmiştir. Firmaların sadece yeni teknolojiyi hızlı bir şekilde değerlere dönüştürmeleri ve yönetmeleri başarı sağlamaları için yeterli değildir. Teknolojik gelişmeler, organizasyonlar arasında rekabeti ve müşteri beklentilerini arttırdığı için firmaları sürekli iyileştirmeye zorlamaktadır.

Yalın düşüncenin çıkış noktası değer kavramıdır. Değer üretici tarafından yaratılır ve sadece kullanıcılar tarafından tanımlanabilir. Firmaların, iş süreçlerini iyileştirebilmek için sürekli çabalamaları, tüm israfı yok ederek, ürün ya da hizmetin tüm aşamalarında değer akışını sağlamaktır. Yalın düşüncenin temelinde değer yaratmak vardır. Yalın düşüncede mükemmellik için sürekli iyileştirme yapılmaktadır.

Sürekli iyileştirme kavramının kökeni, on dokuzuncu yüzyıla ve sanayi devrimi dönemine dayanmaktadır. Verimlilik ve üretkenlik gibi önceliklere insan odağı da eklenerek, sürekli iyileştirme kavramı günümüze kadar gelmiştir.

Bu tez çalışmasının amacı yazılım firmalarında yalın felsefe benimsenerek proje süreçlerinin iyileştirilmesi, hızlı çözüm ve hızlı teslimatla müşteriye sunulmasıdır. Bir Yazılım firmasında uygulama gerçekleştirilmiş ve bulgular değerlendirilmiştir.

Çalışmanın ilk bölümünde süreç ve süreç yönetimi hakkında bilgiler verilmiştir. İkinci bölümde yazılım geliştirme süreci, yazılım projeleri ve yazılım geliştirme modellerinden bahsedilmiştir. Üçüncü bölümde yalın düşünce ve değer akış haritalama tanımlanmıştır. Değer akış haritalama tekniği kullanılarak darboğaz oluşturan süreçlerin tespit edilmesi ve bu darboğaz süreçlerin çözüm alternatiflerinin belirlenerek nasıl bir yol izlenmesi gerektiğine değinilmiştir. Dördüncü bölümde bu konularla ilgili literatür araştırması yapılmıştır. Son bölümde ise önceki bölümlerde açıklanmış olan tüm bilgilerin bir firmanın yazılım departmanında uygulaması yapılmıştır. Bu çalışmada yazılım projeleri yalın tekniklerine ve metodolojilere uygun analiz edilmiştir. Yazılımdaki israf kaynaklarının önlenmesi ve değer katan işlemlerin arttırılarak kaynakların etkin bir şekilde kullanılmasındaki çabalar gösterilmiştir.

2. SÜREÇ YÖNETİMİ VE SÜREÇ İYİLEŞTİRME

Artan rekabet koşullarından kaynaklı kuruluşlarda sürekli gelişim ve iyileştirmeler ihtiyaç haline gelmiştir. Organizasyonların, müşterilerinin istek ve ihtiyaçlarını karşılayabilmek için, süreçlere ihtiyaç duyarlar. Süreç yönetimi ve süreç iyileştirme yöntemleri kavramlarının uygulanması firmaların başarı ve piyasada süreklilik sağlamaları açısından oldukça büyük önem taşımaktadır.

Değişen koşullara uyum sağlamak, sektörde süreklilik sağlamak, üretilen ürünün veya hizmetin kalitesini arttırmak, müşterinin istek ve ihtiyaçlarını karşılayabilmek süreç yönetimini kuruluşun yönetim felsefesi olarak benimsemekle sağlanır.

2.1. Süreç Kavramı

Bir veya daha fazla girdinin firma içi ve/veya dışındaki müşteriler için çözüm oluşturan bir ya da daha çok çıktı şekline dönüştürülmesinin sağlandığı bir faaliyet ya da karşılıklı ilişkileri bulunan faaliyetler kümesine süreç denilmektedir. Belirli bir çıktı başka deyişle ürün ya da hizmet elde etmek için, birbirleriyle etkileşim içerisinde bulunan insanlar, ekipmanlar, malzemeler, yöntemler ve çevresel unsurların toplamıdır (Byrne, 2015).

Süreç; girdileri, çıktıya dönüştüren, tanımlanabilen, sınırları konulabilen, tekrarlanabilen, ölçülebilen, sorumlusu olan, fonksiyonlar arası ve birbirlerine bağlı değer yaratan faaliyetler dizinidir (Şahin, 2002). Kısacası süreç, hedeflenen bir çıktıyı almak için girdiler üzerinde katma değer yaratan işlemlerin bütünüdür. Süreçler, üç temel faaliyetin bileşimidir. Bunlar;

1. Müşteriler için önem taşıyan faaliyetler,
2. Temel olarak fonksiyonel, bölümsel veya örgütsel sınırlar arasında iş akışını sağlayan faaliyetler,
3. Kontrol faaliyetleridir.

Dönüşüm fonksiyonu dört kategoride incelenebilir (Byrne, 2015):

1. ***Fiziksel Dönüşüm:*** Hammadde ya da yarı mamul gibi somut kalemlerin, gerekli bilgiyle birlikte katma değeri daha yüksek olan bir çıktıya çevrilmesidir.
2. ***Konumsal Dönüşüm:*** Fiziksel kalemlerin değiştirilmesidir.
3. ***İşlemsel Dönüşüm:*** Elle tutulamayan, soyut kalemlerdeki değişikliklerdir.
4. ***Bilgisel Dönüşüm:*** Veri değişikliğini ya da verilerin indirgenmesini içerir.

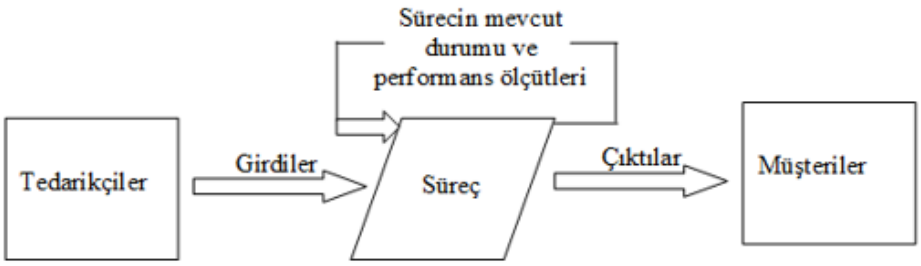
Şekil 1'de görüldüğü gibi Süreç; Girdi, Kaynak, İşlem, Çıktı, Müşteri ve Tedarikçi bileşenlerinden oluşmaktadır.

Şekil 1. Süreç Şeması

Kaynak: Eyüboğlu, (2012)

2.2. Süreç Unsurları ve Bileşenleri

Sürecin temel unsurları Şekil 2'de görüldüğü gibi; girdileri tedarikçi sağlamaktadır. Bu girdileri belli bir süreçten geçerek çıktıyı oluşturmaktadır ve çıktıları da müşteri kullanır. Talepler, bilgiler, hammaddeler ya da yarı mamul maddelere süreç içerisinde ***Girdi*** adı verilir. Müşterinin arzusu taleptir. Süreci başlatan girdilerdir ve süreç adımlarıyla değişim geçiren şeydir.



Şekil 2. Sürecin Temel Unsurları

Kaynak: Eyüboğlu, (2012)

Bir süreçte girdiler; dönüşen kaynaklar ve dönüşümü gerçekleştiren (dönüştüren) kaynaklar olarak ikiye ayrılır. Dönüşen kaynaklar; müşteri, bilgi ve malzemenin bileşimidir. Dönüştüren kaynaklar, personel ve tesis (makine) gibi dönüşen kaynaklara etki eden kaynaklardır. Bununla birlikte, süreçlerin temel iki çıktısı vardır: ürün ve/veya hizmet. Ürün, somuttur, depolanabilir ve taşınabilirdir. Hizmet ise, somut değildir, depolanamaz ve taşınamaz. Genellikle tüketilecekleri an üretilirler (Eren, 2010).

Süreç adımlarının ilerlemesini sağlayacak her türlü donanım veya yazılım, yasalar, formaliteler, prosedürler ya da kalem, defter gibi ürünler ***Kaynak*** adı altında yer alır. Bunlar süreci başlatmazlar fakat süreç adımlarının işlemesinde olması gerekenlerdir. Süreçlerde insan faktörü de girdi sayılmaktadır. İnsan faktörü süreç için yan girdidir onlar süreci yürütürler yani katılımcılardır. Süreç adımlarının, müşteri talep ve beklentilerine göre tamamlanmasıyla değişime uğrayarak ortaya çıkan nihai ürün veya hizmete ***Çıktı*** adı verilir. Yani işlemler sonucunda elde edilendir.

Girdileri temin eden kişi ya da kuruluşlara ***Tedarikçi*** adı verilir. Tedarikçiler organizasyonun içinden ya da dışından olabilir. Dış tedarikçiler firmaya ürün veya hizmet sunulmasında, iç tedarikçiler organizasyon içindeki bütün süreçlerin akışında ürün veya hizmet veren fonksiyonların tedariğinden sorumlulardır.

Sürecin başlamasını sağlayan ve çıktılarını kullanan kişi veya kuruluşlara ***Müşteri*** adı verilir. Süreç aşamalarında görev alan ve çeşitli çıktıları kullanan kişilere ***İç Müşteri*** denir (Şahin, 2002).

Bir süreçte bulunması gereken temel özellikler Şekil 3'te de görüldüğü gibi şu şekilde sıralanabilir (Bozkurt, 2003):

- Tekrarlanabilir olma,
- Tanımlanabilir olma,
- Ölçülebilir olma,
- Katma değer yaratmak,
- Kesinlikle bir sorumlunun bulunması,
- Fonksiyonlar arası yapı,
- Hiyerarşinin tersine yatay organizasyonu gerektirme

***Tekrarlanabilir Olma*:** Süreç tekrarlanabilirdir ve tek seferlik değildir. Yani sürecin birden fazla aynı şekilde işlenmesidir. Süreç döngü halinde sürekli işlemektedir. En ufak bir değişiklik kazanca dönüşebilir.

***Tanımlanabilir Olma*:** Süreç unsurlarının, kontrolünün ve performans ölçümünün net bir şekilde ortaya konması ve belgelenebilir olması özelliğidir. Yani tüm süreçleri belirlemek ve belirlenen bu süreçleri yazılı halde tanımlamak gereklidir.

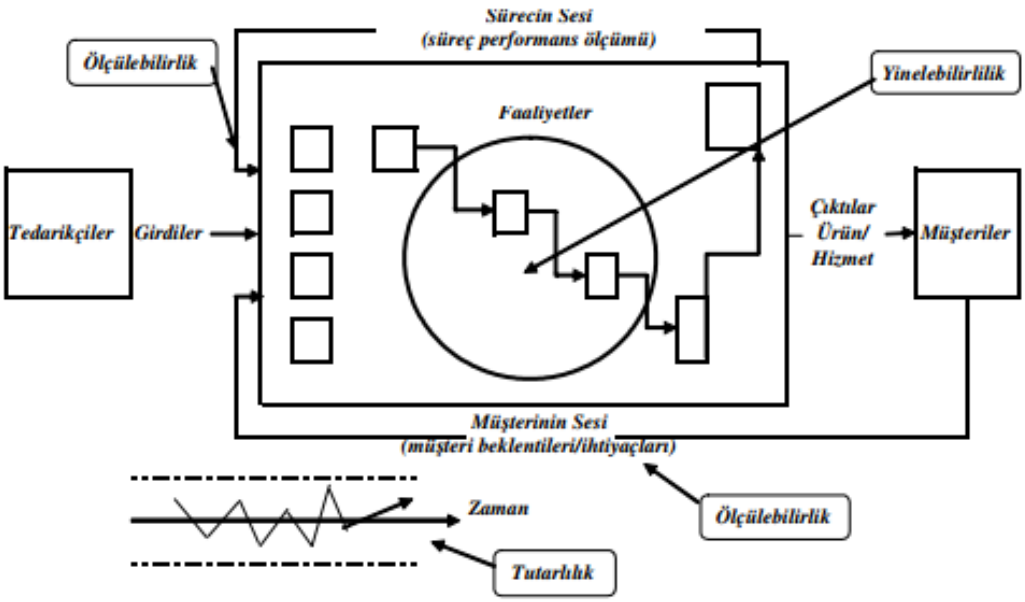
***Ölçülebilir Olma*:** Sürecin kontrol altında tutulabilmesi için ölçülebilir olmalıdır. Ölçülemeyen hiçbir şey kontrol edilemez. İstatistiksel oranlarla sürecin etkililiği ve verimliliği takip edilebilir ve bu böylece sürecin daha iyiye gitmesi için zaman ve maliyet kaybı yaşanmadan planlama yapılabilir.

***Katma Değer Yaratmak*:** Kalite çıktı üreterek müşterinin üzerinde olumlu etki yaratılmasıdır. Operasyonel açıdan süreç, her biri belirli kurallara bağlı ve karşılıklı etkileşim içindeki girdi ve çıktılara sahip olan, sınırları belli iş aktiviteleridir. Süreç bir iş yapma metodudur.

***Kesinlikle Bir Sorumlunun Bulunması*:** Çıktıların müşterinin beklenti ve istekleri doğrultusunda olduğundan sorumlu bir kişinin olması lazımdır. Bu sorumluların sürecin aşamaları ve performansı hakkında her zaman bilgi sahibi olmaları gerekmektedir. Sürecin sorumlusu genellikler lider ya da süreç sahibi olmaktadır.

***Fonksiyonlar Arası Yapının Bulunması*:** Sürecin temelinde arayüz yönetimi vardır. Yani süreç bir noktada başlayıp ve biten faaliyetler dizisi değildir. Süreci oluşturan adımlar arasında ilişki vardır. İş süreci yönetimin amacı bu adımlar arasındaki ilişkiyi fonksiyon olarak tanımlamaktır.

***Hiyerarşinin Tersine Yatay Organizasyonu Gerektirme*:** Organizasyonlarda faaliyetler belirlenip gruplandırılırken geleneksel yönetim sistemi olan hiyerarşinin etkisinden kaçınılmalıdır. İşin yapıldığı yerlerde kararlar alınmalıdır.



Şekil 3. Genel Süreç Şeması

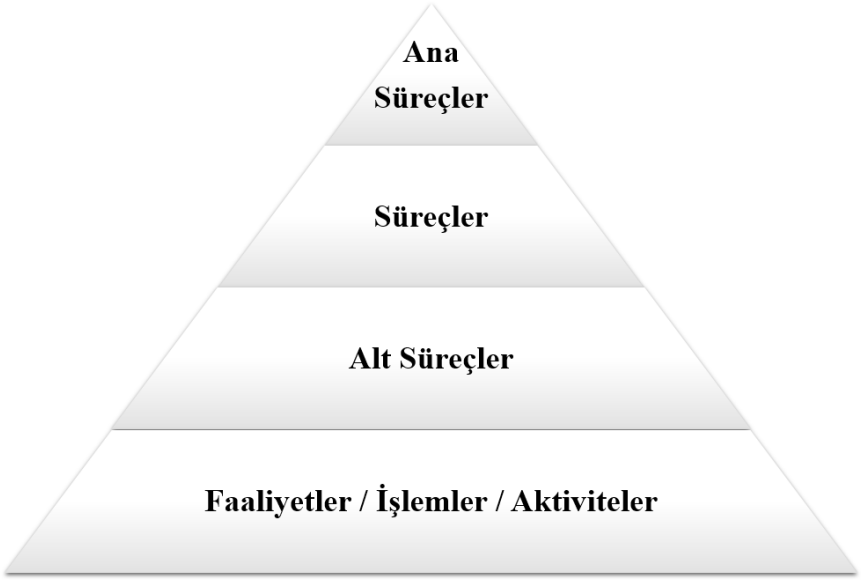
Kaynak: Eyüboğlu, (2005)

2.3. Süreç Hiyerarşisi

Süreç Hiyerarşisi, süreçlerin üst düzey ve alt düzeyleri arasındaki dikey ilişkilerini kapsamaktadır. Süreçlerin yapıları büyüklüklerine göre basit veya karmaşık olabilmektedir. Süreçler bir fonksiyonun içerisinde başlar ve aynı fonksiyonun içerisinde sonlanabilir ya da tüm fonksiyonları kapsayabilir (Eyüboğlu, 2012).

Süreç Hiyerarşisi kapsamı en büyük olandan başlayarak süreçlerin kademeli olarak yapılandırılmasıdır. Bu yaklaşımda Şekil 4'te gösterildiği gibi Ana süreçler, Süreçler, Alt Süreçler ve Faaliyetler/İşlemler/Aktiviteler olmak üzere 4 esas düzey bulunmaktadır (Eyüboğlu, 2005);

- ***Ana Süreçler***; stratejik öneme sahip şirketi direkt etkileyen süreçlerdir. Kuruluşun vizyonu, misyonu ve hedeflerini doğrudan ilgilendirir.
- ***Süreçler,*** ana süreçleri oluştururlar ve birbirleriyle etkileşim halindelerdir.
- ***Alt Süreçler***; süreçleri oluşturan faaliyetlerdir. Bir veya birden fazla fonksiyondan oluşmaktadır.
- ***Faaliyetler / İşlemler / Aktiviteler***; Alt süreçleri oluşturan aynı fonksiyon içerisinde bir ya da birkaç kişi ile birlikte gerçekleştirilen faaliyetlerdir.



Şekil 4. Süreç Hiyerarşisi

Kaynak: Eyüboğlu, (2005)

Süreçler şirket vizyonunu gerçekleştirmede doğrudan etkili olan "Operasyonel Süreçler" ve operasyonel süreçleri geliştirmede yardımcı olan "Yönetsel ve Destek Süreçler" olarak da Tablo 1'deki gibi iki gruba ayrılabilmektedir (Kürkçü, 2005).

Tablo 1. Operasyonel Süreçler ve Destek Süreçleri

Operasyonel Süreçler	**Destek Süreçleri**
Vizyon ve Strateji Geliştirme	İnsan Kaynaklarının Yönetimi
Ürün ve Hizmet Geliştirme	Bilgi Kaynaklarının Yönetimi
Ürün / Hizmet Yönetimi	Enformasyon Yönetimi
Faturalama ve Servis	Çevre Yönetimi
Müşteri Kazanma	Performans Ölçümü
Müşteri İhtiyaçlarının Belirlenmesi	İyileştirme ve Değişim Yönetimi
Sipariş Yönetimi	Varlık Yönetimi
Satış Sonrası Hizmet	Planlama ve Kaynak Ayırma

Kaynak: Kürkçü, (2005)

Süreç belirlemeye genellikle en büyük (makro) süreçlerden başlanılır. Çünkü makro süreçler yönetilebilen mikro alt süreçlere ayrılır. Bir kuruluşun süreçlerini belirlemek için öncelikle kuruluş amacına odaklanılmalıdır. Kuruluşun "Neler yaptığı" ve "Neler yapmak istediği" temel süreçlerine odaklanılmalıdır.

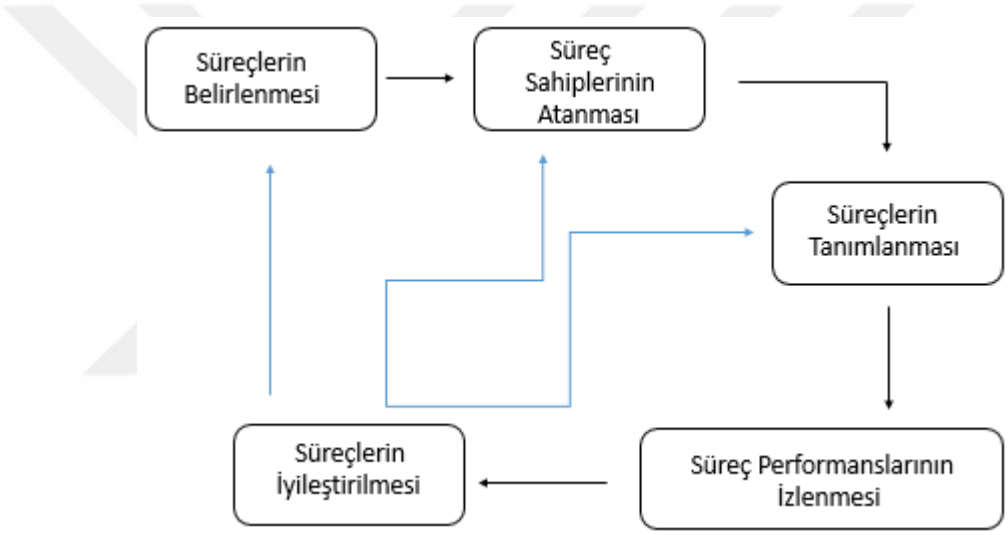
2.4. Süreç Yönetimi

Süreç Yönetimi bir organizasyonun yürütülmesindeki aşamaların tamamlanmasını sağlayan sistematik bir metodolojidir (Eyüboğlu, 2012).

Süreç yönetimi, geleneksel yönetim ürünlerin ya da hizmetlerin kalitesini, performansını ve yarattıkları fayda derecesini kapsamlı olarak ele alan, ürün odaklı bir yönetim şeklidir (Bawden ve Skerritt, 2002). Süreç yönetiminin geleneksel

yönetimden temel farkı; işletmenin fonksiyonlardan ibaret olduğu görüşünün artık savunulmamasıdır. Bu yaklaşım, organizasyon içinde iş akışlarının kolaylaşmasını sağlamaktadır ve istenen bilgilere daha kolay ulaşmak için bir yol oluşturmaktadır.

Geleneksel yönetim tarzında organizasyonlarda müşterilerin ihtiyaçları her zaman ikinci planda olmaktadır. Değer yaratmak sınırlandırıldığından işletmelerin verimlilik ve etkililik oranları düşmektedir. Oranları yükseltmek müşterinin istek ve ihtiyaçları ön planda tutan süreç odaklı yönetim ile mümkün olacaktır (Ataman, 2009). Süreç yönetiminde, süreçlerin devamlı takip edilmesi, geliştirilmesi bir takım adımlarla sağlanmaktadır. Süreç Yönetim adımları aşağıdaki gibidir:



Şekil 5. Süreç Yönetimi Adımları

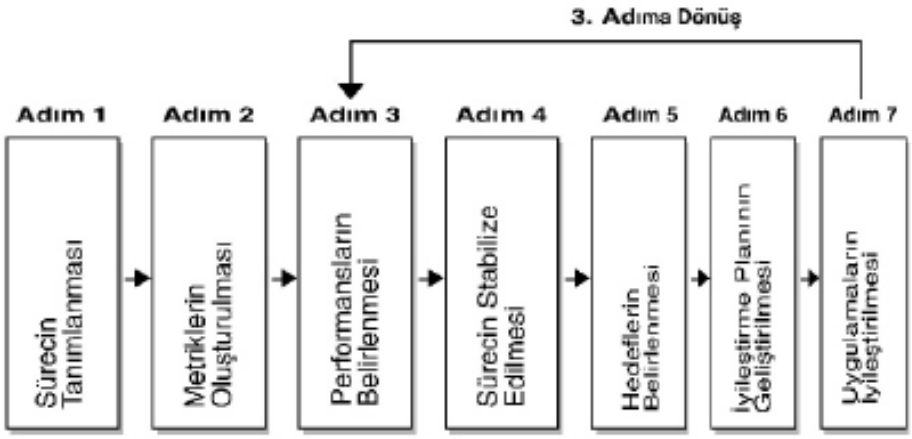
Kaynak: Ataman, (2009)

Günümüzde firmalar devamlı değişmekte ve gelişme olan pazar şartlarına ayak uydurmak için ürün ve hizmetlerin son sunum aşamasına kadar olan süreçlerini düzenli olarak iyileştirmeleri gerekir.

2.5. Süreç İyileştirme

Süreç İyileştirme, varolan süreçleri geliştirmek ve yeni bir düzeye taşımak içindir. Süreçte değişiklikler yapılarak daha etkin ve daha verimli olmasının sağlanmasıdır. Zaman içerisinde her sürecin etkinliğini artırmak ve güncelliğini korumak için düzenli olarak iyileştirme çalışmaları yapılmalıdır. Süreç iyileştirme metodolojisi aşağıdaki adımlardan oluşmaktadır (Eyüboğlu, 2012):

1. Hazırlık
    - ✓ Üst yönetime süreç yönetimi semineri verilmesi,
    - ✓ Süreçlerin ve süreç sahiplerinin belirlenmesi,
    - ✓ İyileştirilecek süreç veya süreçlere karar verilmesi,
    - ✓ İyileştirme ekip veya ekiplerinin oluşturulması,
    - ✓ Ekip üyelerinin ve yedeklerinin eğitim alması,
2. Mevcut Durumun Analizi
    - ✓ Taslak süreç haritası,
    - ✓ Müşterilerle görüşmeler,
    - ✓ Süreçte çalışanlarla görüşmeler,
    - ✓ Sürecin durumunun ve iyileştirilecek alanların ortaya çıkması,
    - ✓ Pareto diyagramları,
    - ✓ Neden sonuç diyagramları,
    - ✓ Histogramlar,
    - ✓ Kontrol tabloları,
    - ✓ Dağılma diyagramları,
    - ✓ Grafikler,
3. Sürecin Çözümlenmesi
    - ✓ Başkalarının neler yaptığının araştırılması,
    - ✓ Hedef koyulması,
    - ✓ Sorunların kökenlerinin tespit edilmesi,
    - ✓ İyileştirme seçeneklerini belirlenmesi,
    - ✓ Seçenekler arasında seçim yapılması,
4. İyileştirmenin Uygulanması
    - ✓ Her seferinde bir ya da iki kritik süreç ele alınarak iyileştirme gerçekleştirilmesi.

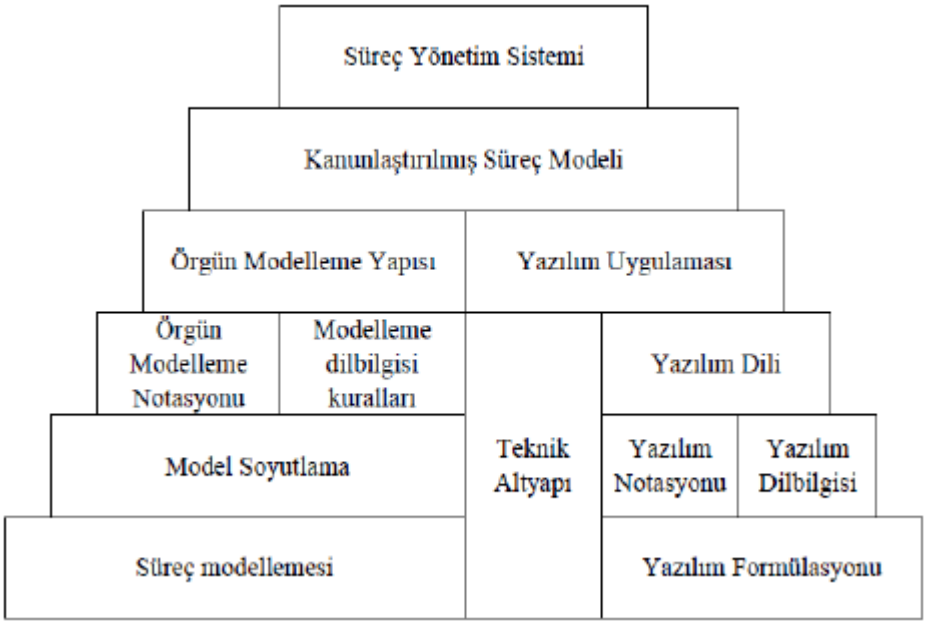


Şekil 6. Süreç İyileştirme Adımları

Kaynak: Ataman, (2009)

Şekil 6'da görülen yedi aşamalı model süreçlerin tanımlanması oluşturmaktadır. Süreçlerin planlanması ve ölçülebilmesi için metriklerin oluşturulması gereklidir. Süreçle ilgili doğru kararların verilebilmesi için performanslar belirlenmelidir. Daha sonraki adımlar üzerinde süreçle ilgili hangi noktaya gelinmek istendiğinin tespiti için iyilştirme planlarının geliştirilmesi ve uygulamaların iyileşitirilmesi gereklidir. Sürekli İyileştirme Süreç Yönetiminin önemli bir parçasıdır. Süreç iyileştirmesi gereken bir firmada, tüm çalışanların katılımı ve devamlılık gereken bir çalışma biçimi olduğu bilinmesi gerekmektedir.

Şekil 7'de görüldüğü gibi Süreç yönetim sistemini süreçlerin modellenmesi, teknik altyapının oluşturulması, yazılımsal formülasyonların belirlenmesi gibi aşamaların tamamının organize, tutarlı ve disiplinli bir şekilde sürekli iyileştirilmesiyle ve düzenli olarak gözlemlenmesiyle meydana gelir.



Şekil 7. Süreç Yönetim Mimarisi

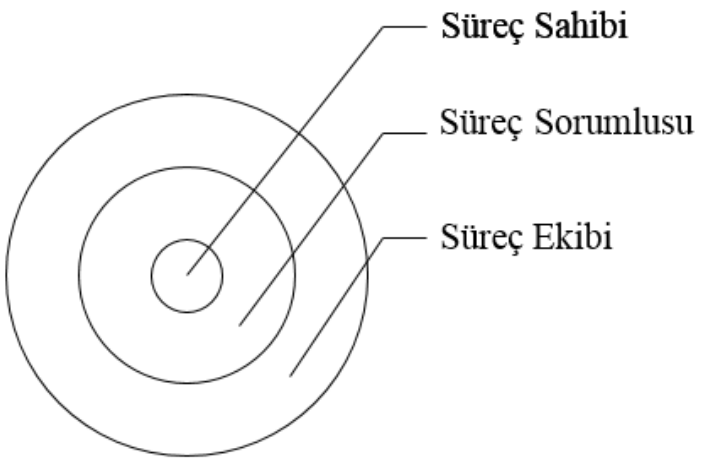
Kaynak: Eren, (2010)

Süreç Yönetimi, iş alanı, büyüklüğü ne olursa olsun bir kuruluşun (Eyüboğlu, 2012):

1. Vizyon, misyon ve hedefleri doğrultusunda süreçlerini belirlemesi,
2. Belirlediği süreçleri tanımlaması,
3. Bu süreçlere süreç sahibi ataması,
4. Süreçlerin performans göstergelerini belirlemesi,
5. Süreç performans göstergeleri için hedefler belirlemesi,
6. Tüm bunları belgelemesi (dokümante etmesi) ki bu günümüzde artık kağıtlar üzerinde değil bilgisayar ortamında olacaktır. Her bir çalışan süreç dokümantasyonuna gereksinme duyduğunda bunlara kolaylıkla erişebilmesi için gerekli bilgi sistemlerinin kurulması,
7. Süreç performans göstergelerini düzenli biçimde ölçmek, izlemek, raporlamak için Ölçüm Sistemi kurması,
8. Süreç göstergeleri hedeflerden kötüye gittiği zaman iyileştirme çalışmalarına başlanmasını sağlaması,
9. İyileştirilmelerin gerçekleştirilmesi ve 7.-9. adımların sürekli bir döngü olarak tekrarlanması anlamına gelir.

Süreç yönetiminin başarılı şekilde yürütülmesi için yukarıdaki maddelerin tamamının uygulanması ve bu maddelerle ilgili kurumdaki tüm çalışanların bilgilendirilmesi ve benimsemelerinin sağlanarak süreç odaklı firma kültürü oluşturulmalıdır. Süreç yönetimi aşamalarında süreç performans göstergeleri sürekli kontrol edilmeli ve sayısal verilerle analiz edilmeli, hedeften sapma durumu olduğunda bunu ilk farkeden kişinin süreç sahibinin olması sağlanarak bu kişinin anında iyileştirme faaliyetlerini başlatması sağlanmalıdır. Süreç iyileştirme başlatılacağı zaman gerekli zaman, insan ve mali kaynakların temini için üst yönetim süreç yönetimini benimsemeli ve bu konuda kararlı olduğunu göstererek her zaman destek olmalıdır. Şekil 8'de belirtilen süreç yönetiminde organizasyon yapısı aşağıdaki gibidir (Eren, 2010):

- ***Süreç Sahibi***: Sürecin tüm aşamalarıyla ilgili bilgisi vardır. Her aşamasıyla süreci bir bütün olarak yönetir. Müşterilerin beklenti ve isteklerinin takibini yapan, süreç sonuçlarından sorumlu olan ve sonuçlara göre analizini yapan kişilerdir.
- ***Süreç Sorumlusu***: Süreç aşamalarında kendi alanlarını yöneten ve organizasyonda direk süreç sahibine bağlı olan kişilerdir.
- ***Süreç Ekibi***: Süreç sorumlusuna bağlı, süreç ve alt süreç aşamalarında geliştirme ve iyileştirme yapan birden fazla kişilerin oluşturduğu topluluktur.



Şekil 8. Süreç Yönetimi Organizasyon Yapısı

Kaynak: Eren, (2010)

2.6. Süreç Yönetiminin Tarihsel Gelişimi

İnsanlar, işletmeler ve yapılan işler her geçen gün daha da gelişir ve müşterilerin beklentileri ve zamanın şartlarına göre değişiklik gösterir. Beklentileri en üst seviyede sağlamak yönetim süreçleriyle olur. Beklentiler değişimdir ve her değişimin bir süreci vardır. Bu süreçlerin yönetimleri beklentilere ve zamanın şartlarına göre dönemsel değişiklik gösterir.

Günümüzde "Süreç Yönetimi" üretim ve iş süreçlerini bir bütün olarak ele almakta ve tüm süreçler değerlendirilmektedir (Aydın, 2007). Yönetim modellerinin evrimleri aşağıdaki gibidir:

- Klasik Yönetim Teorisi
- Neo Klasik Yönetim Teorisi
- Modern Yönetim Teorisi

Klasik Yönetim Teorisi: Öncüleri bilimsel yönetim yaklaşımı ile Frederick Taylor, yönetim süreci yaklaşımı ile Henri Fayol ve bürokrasi yaklaşımı ile Max Weber'dir. Bu döneme bilimsel yönetim dönemi de denir. Bilimsel yönetim dönemi 1880-1930 yılları arasını kapsamaktadır A.B.D.'de gelişme göstermiştir. Frederick W.Taylor, A.B.D'de 1880 yılında kurulan Amerikan Mühendisleri dernek çalışmalarından etkilenerek konferanslar vermek suretiyle bu yönde çalışmalara ağırlık vermiş ve bilimsel yönetimin babası unvanını almıştır.

Neo Klasik Yönetim Teorisi: Klasik yönetim teorisine tepki olarak doğmuştur. Neo Klasik Yönetim teorisinin öncüsü Elton Mayo'dur.

Neo Klasik Yönetim düşüncesinin esasında insan ilişkileri yaklaşımı oluşturmaktadır. İnsan davranışı, insanlar arası ilişkiler, gruplar ve davranışları, algılama ve tutumlar, motivasyon, liderlik, organizasyonlarda gelişme ve değişme konularına odaklıdır (Ataman, 2009).

***Modern Yönetim Teorisi*:** II. Dünya Savaşı'ndan sonra 1950'li yıllarda başlayan ve günümüze kadar devam eden analitik temele dayalı, model ve sistem

kurmaya yönelik yaklaşımdır. Modern Yönetim Teorisi diğer iki teorinin de incelemede yetersiz kaldığı örgütleri ele alarak, örgütü bütünsel olarak inceleyen örgüt-çevre etkileşiminin üzerinde duran bir teoridir. Modern Yönetim Teorisi'nin temeli iki yaklaşımdan oluşmaktadır (Koçel, 2001). Bunlar:

- Sistem Yaklaşımı
- Durumsallık Yaklaşımı

Sistem Yaklaşımı; bir ya da birden çok amaca ya da sonuca ulaşmak için aralarında ilişkiler bulunan fiziksel ya da kavramsal, birden çok bileşenin oluşturduğu bütüne sistem denir. Sistemi oluşturan üç ana noktalar parçaların birbirleriyle ilişkili, uyumlu ve çalıştıklarında bir bütün oluşturmasıdır. Hem tüm parçalar hem de parçalardan oluşan bütün büyük önem taşır. Analitik temele sahip bir teoridir, alt birimlerden oluşur ve iç ve dış çevre ile etkileşim halindedir. Sistem yaklaşımı hem makro hem de mikro yaklaşımdır (Koçel, 2001).

Sistem Yaklaşımı'na damga vuran kuramcılar (Aydın, 2007):

- Alfred Korzybski (Gerçekliğin süreci),
- Mary Parker Follet (Eşgüdüm),
- Chester Barnard (İşbirliğine dayalı dinamik bir sistemin kontrolü)
- Ludwig Von Bertalanffy (Örgütlerde çok yönlü dinamik bir etkileşim olasılıklarının dikkate alınması, örgütlerde alt birimlerin ve bu alt birimlerin oluşturduğu bütünün önemi)

Durumsallık Yaklaşımı; sistem yaklaşımından hareketle bir dizi araştırma yapılarak durumsallık yaklaşımı geliştirilmiştir. Durumsallık teorisi hakkında ilk araştırma 1965 yılında Woodward tarafından yapısal değişikliklerin ekonomik başarıyı ne kadar etkilediğini saptamak için İngiltere'nin 100 firmasında incelemesiyle başlamıştır. Durumsallık yaklaşımına göre yöneticilerin benimseyeceği hiçbir evrensel yönetim ilkelerinin ya da tekniklerinin bulunmayacağı kabul görülmüştür (Aydın, 2007).

3. YAZILIM GELİŞTİRME SÜRECİ

Yazılım ürünleri, hayatımızın her alanında kişilerin ve kurumların ihtiyaçlarını karşılamada çeşitli çözümler sunmaktadır. İnternet, bilgiye ve hizmete her yerden erişebilirlik sağlayarak ticaret, eğitim ve sosyal ilişkilerde köklü değişimlere aracılık etmektedir. Yazılan bir program başka bir programın yazılmasına da yardımcı olmaktadır. Veritabanları ve işletim sistemleri diğer yazılımlar için altyapı sunmaktadır.

Yazılım bir teknoloji terimidir. Teknolojiyle iç içe olan her alanda sistemleri oluşturan, donanım parçaları dahil olmak üzere yöneterek komut veren, o donanımı kullanan kişiye gerekli komut dosyaları ile cevap vererek kullanmasını sağlayan birime ***yazılım*** denir (Svensson, 2006).

Yazılım geliştirme, ana hedef ve kapsamın tespit edilmesi ile başlayan, planlama, analiz, tasarım, gerçekleştirme, test, devreye alma ve bakım aşamalarından oluşan bir süreçtir. Süreç ilerledikçe ve bilgiler netleştikçe aşamaları tekrarlayabilir. Yazılım ürünü geliştirme aşamasında gösterilen dikkat ve özen, yazılım geliştirmede kullanılacak süreç belirlenirken de gösterilmelidir (Svensson, 2006).

Proje yöneticisi, yazılım ekibinin de katkılarıyka yazılım geliştirme sürecini planlar ve yönetir. Yöneticinin konuya hakim olması, sürecin her aşamasında yapılan çalışmaları bilmesi, çıkabilecek sonuçları tahminlemesi gerekmektedir. Yazılım geliştirme süreci kişi, kurum, proje yapılarından kaynaklanan farklı istek ve ihtiyaçlara göre şekillenmektedir. Bu durum doğrusal, yinelemeli ve çevik geliştirme gibi süreç yönetim modellerinin ortaya çıkmasına neden olmuştur (Nizam, 2015).

3.1. Yazılım Projeleri

Yazılımın hizmet verdiği alanlar ile kaliteli yazılım beklentisi doğru oranda artmaktadır. Kurumlar arası rekabet, teknolojik gelişmeler, müşteri istek ve

ihtiyaçlarının değişimi mevcut süreçlerin iyileştirilmesini tetiklemektedir. Bu tür iyileştirmeler için kapsam belirlemek, plan ve ekip oluşturmak, oluşturulan plana bağlı çalışarak işleri belirlenen süre çerçevesinde tamamlanması gereklidir. Bu süreçlerin tamamına da proje denilmektedir (Nizam, 2015).

Kısaca tanımlandığında proje; belli bir başlangıç ve bitiş tarihi olan, amacı ve bütçesi net bir şekilde belirlenmiş bir defaya mahsus gerçekleştirilecek aktivitelerin bütünüdür.

Yazılım projesi; kişi veya kurumun belirli bir ihtiyacını analiz ederek elektronik ortamda karşılamak için geliştirilen, belli bir geliştirme süresi olan yazılım geliştirme çabasıdır. Proje genellikle zaman, maliyet ve kapsam değişkenleriyle özdeşleştirilir. Tüm projelerde yapılacaklar, maliyet ve zaman kısıtları bulunmaktadır. Maliyet, projenin gerçekleştirilmesi için gerekli tüm kaynakların maliyetidir. Kapsam, ürün ve proje kapsamı şeklinde iki temel parçadan oluşmaktadır. Ürün kapsamı, istenilen özellik, işlev ve kalite şartlarından oluşur. Proje kapsamı istenilen ürünü teslim etmek için yapılacak iş, görev ve aşamalardır (Nizam, 2015).

Tablo 2'de görüldüğü gibi projenin yaşam çevrimi planlama, geliştirme, uygulama ve sonuçlandırma olmak üzere dört ayrı evreden oluşur.

Tablo 2. Proje Evreleri

	Planlama	Geliştirme	Uygulama	Sonuçlandırma
Amaçlar / Hedefler	* Gerçek sorunu belirleyin * Yarardaşları belirleyin * Proje hedeflrini ve amaçlarını belirleyin	* Ekibi bir araya getirin * Genel planı geliştirin	* Süreci gözleyip kontrol edin * Kaydedilen ilerlemeyi rapor edin	* Projeyi sonlandırın * Sonraki adımları tespit edin
Faaliyetler	* Kapsamı, belli başlı faaliyet ve görevleri belirleyin * Çaba ve süre tahmininde bulunun * Kaynak ihtiyaçlarını belirleyin * Seçenekleri hazırlayın	* Zaman planlaması yapın * Kritik yolu çizin * Ekibi motive edin * Görevlere insan atayıp kaynak tahsis edin * Bütçeyi hazırlayın * Görevleri ihtiyaca göre başkalarına devredin * Yarardaşların beklentilerini netleştirin	* Yürüyen çalışmayı değerlendirip onaylayın * Projenin kilometre taşlarını açıklayın * Gelişme sürecini yönetin * Kaydedilen ilerlemeyi ve sorunları yarardaşlara bildirin	* Performans değerlendirmesi yapın * Pojeyi kapatın * Çıkan dersleri ekiple paylaşın * Bir izleme planı hazırlayın * Sonuçları yarardaşlarla birlikte değerlendirin
Kilit Beceriler	* Görev analizi * Planlama * Tercihlere ilişkin maliyet - yarar analizi	* Süreç analizi * Ekip oluşturmak * Başkalarına görev devretmek * Müzakere * İşe insan atamak ve almak * İletişim	* Nezaret etmek * Liderlik yapmak ve motive etmek * İletişim * Anlaşmazlık yönetimi * Sorun çözmek	* İzleme * Planlama * İletişim
Araçlar	* İş ayrıştırma yapısı (İAY) * Beceri envanteri	* Zaman planlaması araçları (CRM, PERT, GANTT)	* Kaydedilen ilerlemeyi raporlama araçları	* Proje değerlendirmesi araçları

Kaynak: Çandur, (2010)

3.2. Yazılım Proje Yönetimi

Proje yönetimi, planlanmış bir yaklaşımın olmadığı durumlarda kişilerin projeleri başarıyla sonuçlandırılamadığının görülmesiyle geliştirilmiştir. Proje yönetiminin amacı, projelerin tamamlanması, belirli maliyet ve planlanan süre içerisinde nihai noktaya ulaşmasıdır (Newton, 2015). Projeler yararlı bir değişim gerçekleştirmek ve varolan duruma değer katmak için oluşturulur. Her bir projenin esneklik payını da hesaba katarak belirli başlangıç ve bitiş tarihleri mevcuttur.

Proje Yönetimi'nin ilk uygulaması çoğu literatürde Eski Mısır zamanında yapılan Piramitler kabul edilmektedir. Fakat bu piramitlerin hangi planla ve nasıl yapıldığına dair hala çok kesin bilgiler bulunmamaktadır. İlk Proje Yönetimi danışmanı Amerikalı iktisatçı ve mühendis Frederick Taylor (1856-1915)'dır. Taylor yönetim tekniklerinin bilimsel olarak analiz edilip geliştirilebileceğini göstermiştir.

Çalışmaları verimliliği artırmaya odaklıdır. İş süreçlerinin bileşenlerini ayrıntılı bir şekilde analiz etmiştir (Nizam, 2015).

Modern Proje Yönetimi'nin öncüsü ise Henry Gantt (1861-1919)'tır. Günümüzde yaygın olarak kullanılan proje izleme ve değerlendirme yöntemlerinden Gantt şemalarını geliştirmiştir. Bu şema proje takvimini oluşturmada büyük kolaylık sağlamaktadır (Çandur, 2010).

Yazılım proje yönetimi; müşterinin istek ve ihtiyaçlarının verilen kaynaklarla, belirlenen süre ve maliyet içerisinde tamamlanması için gerekli planlama, yönetim, koordinasyon, bilgilendirme, gözden geçirme ve çıkan sorunları çözmek için gerekli düzeltme ve iyileştirmeleri yapma faaliyetlerinin tümüdür. Büyük projeler için ekip olarak çalışmak zorunludur. Proje yönetimi, ekip çalışmalarını amaç bütünlüğü kapsamında yürütme, ekip içerisinde iletişimi sağlama, ekibi yönlendirme ve yapılan çalışmaların amaçlara uygunluğunu değerlendirme faaliyetleridir.

Nizam (2015)'a göre bir yazılım projesinde ekip üyeleri Şekil 9'da görülen; yazılım proje yöneticisi, yazılım takım lideri, yazılım mimari, sistem (iş) analisti, yazılım geliştirici, arayüz geliştirici, ürün yöneticisi, kalite (test) mühendisi, yardımcı ürün geliştirici, risk yöneticisi, son kullanıcı belge geliştirici, destekleyici/sponsor ve müşteri/son kullanıcıdan oluşmaktadır.

3.2.1. Yazılım Proje Yöneticisi

Projenin istenilen kalitede ve sürede tamamlanması için gerekli planın hazırlanması, standartların belirlenmesi, teknik altyapı seçimi, plandaki faaliyetlerin koordinasyonu, görevlere uygun kişinin atanması ve ekibin plana uygun hareket etmesi kontrolünden sorumlu olan kişidir. Temel kararlar alır.

Yazılım Proje yöneticisi kullanıcı ihtiyacı, üst yönetim beklentisi ve yazılım ekibi tecrübesi arasında köprü vazifesi yapar. Projedeki önemli kararları verir, planı oluşturur, kalite standartlarını belirler, değişimi yönetir, bilgilendirme ve iletişim faaliyetlerini yürütür, ekibin moral ve motivasyonunu sağlar. Proje yöneticisi

projelerin müşteri ve son kullanıcı için yapıldığını ekibe hatırlatmalı, müşteriye bir adım yakın durmalıdır. Kurum üst yönetimi de özel durumlar hariç proje yöneticisinin kararlarını desteklemelidir. Teknik kararlar konunu uzmanı kişiler tarafından alınmalıdır (Nizam, 2015).

3.2.2. Yazılım Takım Lideri

Proje yöneticisine bağlı olarak çalışan teknik yöneticidir. Projenin teknik altyapısının plana göre geliştirilmesinden sorumludur. Proje yöneticisiyle birlikte teknik gereksinim arasında bağlantı oluşturur. Küçük ve orta ölçekli projelerde genellikle proje yöneticisi ile takım lideri aynı kişidir.

3.2.3. Yazılım Mimarı

Yazılım Mimarı kullanıcı ihtiyaçlarını karşılamak için analiz, tasarım ve gerçekleştirilme aşamalarını kapsayarak yazılım mimarisini oluşturan kişidir. Organizasyonun sistemsel yapısı, arayüzleri, yapısal ve davranışsal elemanlarının alt sistemlere dağılımı hakkında kararlar verir. Oluşturulan yazılım mimarisi model için yol göstereceğinden Yazılım Mimarı'nın tecrübeli olması gerekir.

3.2.4. Sistem Analisti (İş Analisti)

Sistem Analisti iş biriminin ihtiyaçlarını ve isteklerini analiz ve tespit ederek bunlarla ilgili detaylı doküman hazırlayan kişidir. Dokümanın yazılım aşamasında önemli olduğundan hazırlamadan önce iş biriminin isteklerini iyi anlamak gerekir ve hazırlama aşaması işbiriminin kontrolünde olmalıdır.

3.2.5. Yazılım Geliştirici

Yazılımın gerçekleştirilmesinden sorumlu kişidir. Veritabanında kod seviyesinde sistem analistinin hazırladığı gereksinim analizine göre geliştirme yapar. Kullanacağı kod dilini ve algoritmaya kendi karar verir. Modelin oluşumu onun sorumluluğundadır.

3.2.6. Arayüz Geliştirici

Yazılıma erişim sağlanan arayüzlerin tasarımından sorumludur. Görsel dizaynı işbiriminin isteğine göre oluşturur. Veritabanındaki kodlarla tutarlı ve uygun geliştirme yapar.

3.2.7. Ürün Yöneticisi

Son kullanıcının ihtiyaçlarından yola çıkarak ve piyasadaki benzer ürünleri inceleyerek yazılımın ticari bir ürün olarak gelecekteki yerini belirler.

3.2.8. Kalite / Test Mühendisi

Yazılımdan sonraki aşamada kullanıcı isteklerine göre uygun olup olmadığına dair kontrolünü sağlarlar. Test aşaması hata bulmaya yöneliktir. Sürecin vakit kazanma açışından en önemli kısımlarından biridir. Hata erken farkedilip yazılım geliştirme kısmına iletilerek müdahale edilir.

3.2.9. Yardımcı Ürün Geliştirici

Projenin geliştirilmesinde işlerin hızlandırılması için kullanılacak ürünlerin teminini ve organize edilmesini sağlayan kişidir. Süreçte zaman kaybını önlemeye yönelik çalışmaları olur.

3.2.10. Risk Yöneticisi

Projede karşılaşılması muhtemel olumsuzları tespit ve takip ederek durumu aşarak sürecin devamını sağlama ile ilgili çalışmalar yapan kişidir. Projelerin amaçlarından biri de hedef tutturma olduğundan bu durum için Risk Yöneticisi'nin sorumluluğu oldukça büyüktür.

3.2.11. Son Kullanıcı Belge Geliştirici

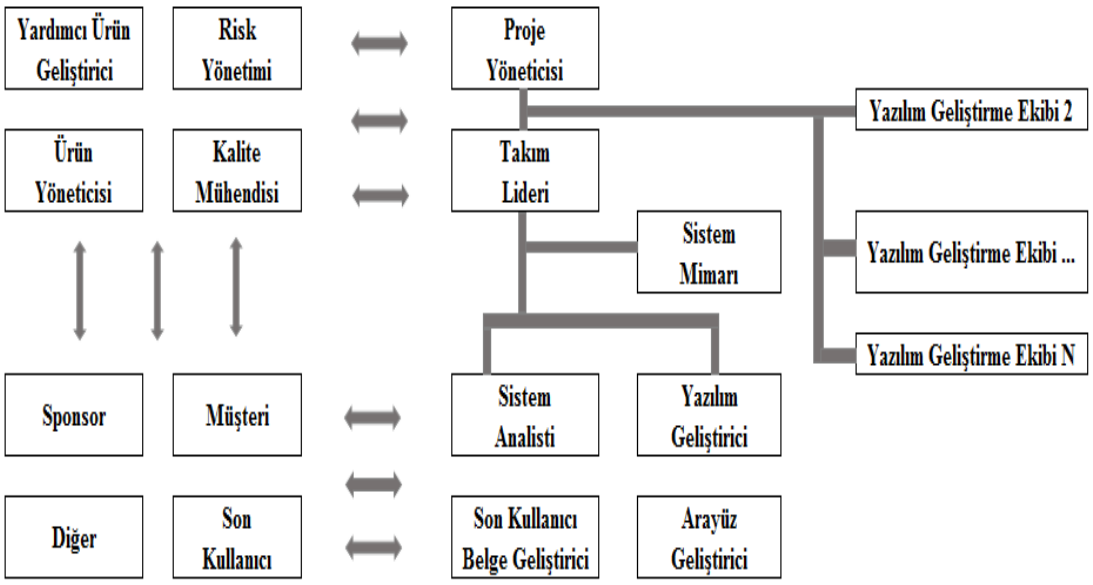
Son kullanıcıların, oluşturulan ürünü kullanabilmeleri için hazırlanan dokümandır. Bu dokümanlar kullanıcı eğitimlerinde de kullanılır. Kullanıcının anlayacağı bir dilde yazılmalıdır. Bu dokümanları genellikle iş analistleri hazırlar.

3.2.12. Destekleyici / Sponsor

Proje için idari destek ve mali kaynak sağlayan kişidir. Projenin ayrıntılarıyla değil geneliyle ilgilenir.

3.2.13. Müşteri / Son Kullanıcı

Proje sonunda ürünü kullanacak kişidir. Ortaya çıkacak ürünün beklentilerini ve isteklerini karşılaması için yazılım aşamalarının çoğunda olması gereklidir.



Şekil 9. Yazılım Ekibinde Roller Arası İletişim

Kaynak: Nizam, (2015)

3.3. Yazılım Projelerinde Yaşanan Problemler

Son yıllarda Proje yönetiminin BT alanında kullanımı yaygınlaşmıştır. Fakat Proje yönetiminin yaygınlaşmasıyla projelerin başarısızlık yüzdelerinin yüksek olduğu belirlenmiştir ve bu durum BT proje yönetiminin nasıl olması gerektiği konusundaki ihtiyacın gittikçe artmasına sebep olmuştur.

2014 yılında Standish Group tarafından yapılan ankete göre (Hastie, 2015) proje seçimini etkileyen faktörler ve oranları Tablo 3'teki gibidir:

Tablo 3. Proje Seçimini Etkileyen Faktörler ve Oranları

1. Kullanıcı Giriş Hatası	**% 12.8**
2.Eksik Gereksinimler ve Spesifikasyonlar	**% 12.3**
3.Değişim Gereksinimleri ve Spesifikasyonları	**% 11.8**
4.Yönetici Desteği Eksikliği	**% 7.5**
5.Teknoloji Yetersizliği	**% 7**
6.Kaynak Yetersizliği	**% 6.4**
7.Gerçekçi Olmayan Beklentiler	**% 5.9**
8.Belirsiz Hedefler	**% 5.3**
9.Gerçekçi Olmayan Zaman Çerçeveleri	**% 4.3**
10.Yeni Teknoloji	**% 3.7**
Diğer	**% 23**

Kaynak: Hastie, (2015)

Aynı yıl araştırmasına göre projelerin başarılı olmasını etkileyen faktörler ve oranları Tablo 4'teki gibidir:

Tablo 4. Projelerin Başarılı Olmasını Etkileyen Faktörler ve Oranları

1. Kullanıcı Katılımı	**% 15.9**
2. Üst Düzey Yönetim Desteği	**% 13.9**
3. Şartların Açıkça Belirtilmesi	**% 13**
4. Uygun Planlama	**% 9.6**
5. Gerçekçi Beklentiler	**% 8.2**
6. Daha Küçük Projelerin Kilometre Taşları	**% 7.7**
7. Yetkili Personel	**% 7.2**
8. Sahiplik	**% 5.3**
9. Net Vizyon ve Hedefler	**% 2.9**
10. Yoğun ve Odaklanmış Çalışanlar	**% 2.4**
Diğer	**% 13.9**

Kaynak: Hastie, (2015)

Projelerin tamamlanmasına engel olan iptal edilmelerini sağlayan faktörler ve oranları Tablo 5'deki gibidir:

Tablo 5. Projelerin Tamamlanmasına Engel Olan İptal Edilmelerini Sağlayan Faktörler ve Oranları

1. Eksik Gereksinimler	**% 13.1**
2. Kullanıcı Katılımının Yetersizliği	**% 12.4**
3. Sorunlu Kaynaklar	**% 10.6**
4. Gerçekçi Olmayan Beklentiler	**% 9.9**
5. Yürütme Desteğinin Yetersizliği	**% 9.3**
6. Değiştirme Gereksinimleri ve Spesifikasyonları	**% 8.7**
7. Planlama Eksikliği	**% 8.1**
8. İhtiyaç Fazlalıkları	**% 7.5**
9. BT Yönetiminin Yetersizliği	**% 6.2**
10. Teknoloji Okur Yazar Olmadığı Durum	**% 4.3**
Diğer	**% 9.9**

Kaynak: Hastie, (2015)

Yazılım projelerindeki başarısızlık hem maddi hem de manevi büyük kayıplara yol açmaktadır. Yazılım projelerini başarısızlığa götüren sebepler aşağıdaki gibidir (Hastie, 2015):

- Proje yöneticilerinin kullanıcı gereksinimlerini anlayamaması,
- İstek ve ihtiyaçlarının eksik ya da yanlış tanımlanması,
- Proje değişiklerinin doğru yönetilememesi,
- Seçilen teknolojinin değişmesi,
- Proje teslim tarihinin gerçekçi olmaması,
- Kullanıcıların memnuniyetsizliği,
- Proje yönetiminin önceden edinilmiş tecrübelerin dikkate alınmaması,
- Projeye uygun ekip çalışanlarının seçilememesi.

Günümüzde yazılım projelerinin birçoğu başarılı bir şekilde tamamlanamamaktadır. Standish Group Bilişim Teknolojileri alanında yaşanan projelerdeki başarısızlıklarla ilgili düzenli çalışmalar yürütmektedir. Her sene yayınladıkları 'Chaos' raporlarında (Hastie, 2015) projeleri başarılı, zorlanmış ve başarısız olarak sınıflandırmışlardır. Standish Group raporuna göre: Amerika Birleşik Devletleri'nde Bilişim projelerine her yıl 250 milyar dolar harcama ile yaklaşık 175.000 proje geliştirilmektedir. Bir gelişmenin ortalama maliyeti büyük şirket için 2.322.000 $, orta büyüklükte bir şirket için 1.331.000 $ ve küçük bir şirket için 434.000 $'dır. Yazılım projelerinde her zaman kaos vardır. Standish Group 1985 yılında kurulan kamu ve özel sektörde bilgi sistemleri uygulama projeleri konusunda raporlar bildiren bağımsız bir uluslararası bilişim araştırma danışmanlık şirketidir. Firma özellikle BT projelerindeki kritik yazılım uygulamaları üzerinde durarak, başarısızlıklara ve olası gelişmelere odaklanır. Herhangi bir planlı projeyi nasıl yöneteceğinizle ilgili ayrıntılar, bir projeyi başarılı bir şekilde uygulamak için gereklidir. Bu, özellikle projelerin modellenmesi için geçerlidir, çünkü burada proje koordinasyonunun klasik görevleri metodolojileri tanımlayarak ve öngörerek tamamlamaktadır. Genel olarak, projeler zaman, karmaşık ve genellikle disiplinler arası olarak sınırlı olan görevleri temsil eder. Bir yazılım projesinin başarılı bir şekilde uygulanması, çalışanın görevleri, kaynakların organize edilmesini, planlanmasını ve kontrol edilmesini gerektirir. (Neumann, Probst ve Wernsmann,

2003). 2015 yılı raporu dünya çapında 50.000 yazılım projesini incelemiştir. Yıllara göre oranları Tablo 6'daki gibidir (Hastie, 2015):

Tablo 6. 1994-2015 Yılları Arası Yazılım Projelerinin Başarı Oranları

	1994	1996	1998	2000	2002	2004	2006	2009	2011	2012	2013	2014	2015
Başarılı	16%	27%	26%	28%	34%	29%	35%	32%	37%	27%	31%	28%	29%
Sorunlu	53%	33%	46%	49%	51%	53%	46%	44%	42%	56%	50%	55%	52%
Başarısız	31%	40%	28%	23%	15%	18%	19%	24%	21%	17%	19%	17%	19%

Kaynak: Hastie, 2015

Tablo 6 incelendiğinde başarısız proje oranının 2002 yılına kadar düştüğü ve sonra yeniden yükseldiği görülmektedir. Başarılı proje oranına bakıldığında gelişen teknoloji, artan bilgi birikimi, kullanılan araç ve tekniklere rağmen oldukça düşük oranlara sahiptir. Projelerin başarılı olabilmesi için en iyi uygulama ve sistemlerinin organizasyon içerisinde uygulanması gereklidir. Projelerin günümüz rekabet sistemine uygun bir şekilde yönetilmesi gerekmektedir. Tablo 7'ye göre ise küçük projeler büyük projelere oranla daha başarılıdır.

Tablo 7. Yazılım Projelerinin Büyüklüğü ve Başarı Oranları

Projenin Büyüklüğü	Başarılı	Sorunlu	Başarısız
Çok Büyük	2%	7%	17%
Büyük	6%	17%	24%
Orta	9%	26%	31%
Orta-Küçük	21%	32%	17%
Küçük	62%	16%	11%

Kaynak: Neumann, Probst ve Wernsmann, (2003)

Yazılım projelerinde özen ve beklenti ile başlatılır ve çoğunlukla yaşam döngüsü tamamlanamadan başarısızlıkla sonuçlanır. Bu başarısızlık birçok faktöre bağlıdır ama en önemlisi proje yönetimindeki eksiklikler ve yanlışlıkların bu sonuca ulaşmadaki payıdır. BT projelerinin en sık karşılaşılan sorunlarından biri zamanında ve bütçe dahilinde bitirilememesidir (Neumann, Probst ve Wernsmann, 2003).

Standish Group tarafından yapılan araştırmalara göre, çalışan sayısının çok olduğu organizasyonlardaki projelerin %61.5'i planlanan bütçeyi aşmakta, tahmin edilen süreden daha geç bitirilmekte ve analizde belirtilen kriterler ve özelliklerin daha azını ortaya çıkarmaktadır (Svensson, 2006). Projelerde ortalama bütçe aşımı %189, zaman aşımı ise %222 olduğu tespit edilmektedir. Tamamlanan projeler, sözleşmede belirtilen kriter ve özelliklerin %67'sini kapsamaktadır. Bu sorunların nedeni bilişim projelerinin kapsamı ve hızı diğer projelere göre farklılıklarıdır. Teknolojinin devamlı gelişmesi, iş süreçlerinin yeniden yapılandırılması, kapsamın genişletilmesi bütçeyi olumsuz yönden etkilediğinden riski de arttırmaktadır.

Maliyet aşırılıkları, zaman aşımları projelerin başarısızlıkla sonuçlanmasını sağlayan en önemli etkenlerdir. Projelerde, neredeyse 150-200% maliyet aşımı olmaktadır. Ortalama tüm şirketlerde maliyet tahmini % 189'dur. Ortalama maliyet büyük şirketler için % 178, orta ölçekli şirketler için % 182 ve küçük ölçekli şirketler için % 214'tür (Svensson, 2006).

Tablo 8. Projelerde Maliyet Aşırılık Oranları

Maliyet Verileri	Oranlar
20% ve altı	**%15.5**
21 - 50%	**%31.5**
51 - 100%	**%29.6**
101 - 200%	**%10.2**
201 - 400%	**%8.8**
401% ve üstü	**%4.4**

ABD'de her yıl BT uygulaması için 250 milyar dolardan fazla harcama yapılmaktadır. Yaklaşık 175.000 projenin geliştirilmesi için büyük bir şirket için ortalama maliyet proje 2.322.000 dolar, orta ölçekli bir şirket için 1,331,000 dolar, küçük bir şirket için ise 434.000 dolar olmaktadır. Bu projelerin birçoğu başarısız olmaktadır. Standish Group araştırmasına göre, projelerin% 31,1'i tamamlanamadan iptal edilmektedir. Diğer sonuçlar, projelerin % 52,7'sini maliyet tahminlerinin % 189'una mal olmaktadır. Kaybedilen fırsat maliyetleri ölçülebilir değildir, ancak trilyonlarca dolar kolayca olabilmektedir. Başarı tarafında, ortalama yazılım projeleri için sadece % 16.2'si zamanında ve tahminlenen bütçede tamamlanmıştır. Büyük şirketlerde, projelerinin yalnızca % 9'u zamanında ve tahminlenen bütçeye uygun

olarak tamamlanmıştır. Küçük şirketlerin tamamlanma oranları büyük şirketlere istinaden daha iyi durumdadır. Yazılım projelerinin toplam % 78.4'ü orijinal özelliklerinin ve işlevlerinin en az% 74.2'siyle sonuçlanmaktadır (Hastie, 2015).

3.4. Yazılım Geliştirme Süreci

Yazılım geliştirme süreci; organizasyonda temel hedef belirlenerek, planlama, analiz, tasaım, kodlama,test, devreye alım ve bakım aşamalarından oluşan bir süreçtir (Nizam, 2015).

İhtiyaçların yazılıma dönüşmesi Şekil 10'da görüldüğü gibi aşama aşama olarak gerçekleşir. İlk olarak yazılımın genel çerçevesi çizilir ve genel bir plan hazırlanır. Sonra ihtiyaçlar ayrıntılı bir şekilde analiz edilir ve bu analizin yazılım araçlarıyla nasıl gerçekleştirileceği tasarlanır. Ekran, rapor, nesne ve veritabanı yapıları, kodlarla oluşturulur. Projenin ihtiyaçları karşılama derecesi ve yazılımda hata olup olmadığı test edilir. Testte bulunan hatalar çözüldükten sonra eğitim ve kurulum aşamasına geçilir. Proje müşteri kullanımına açıldıktan sonra yeni kullanıcı isteği, gözden kaçan hatalar için bakım ve destek çalışması devam eder. Kısaca yazılım geliştirme süreci mühendislikte; Planlama, ihtiyaç analizi, tasarım, gerçekleştirme, test, devreye alma ve bakım şeklinde standartlaşmıştır.

Şekil 10. Yazılım Geliştirme Döngüsü

Kaynak: Nizam, (2015)

Prosesleri farklı olmasına rağmen, bütün yazılım süreçlerinde ortak olan aktiviteler bulunmaktadır (Nizam, 2015). Bunlar:

1. Yazılım özelliklerinin belirlenmesi; yazılım ihtiyaç ve kısıtlarının saptanması,
2. Yazılımın tasarımı ve gerçekleştirilmesi,
3. Yazılımın geçerliliğinin sağlanması ve teslim edilmesi; müşteri ihtiyaçlarının karşılanıp karşılanmadığının test edilmesi ve teslimi,

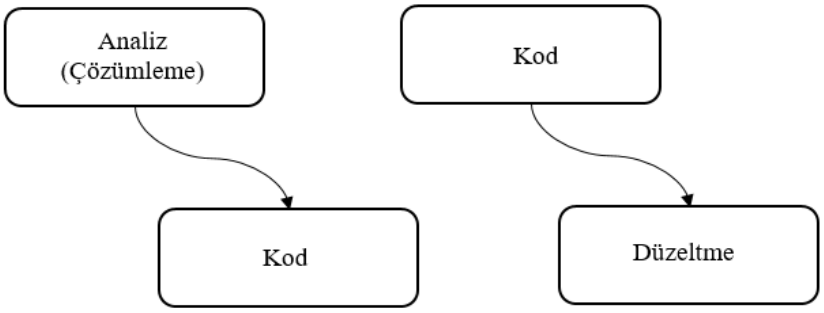
4. Yazılım bakımı; müşteri ihtiyaçlarının değişikliğine göre ya da oluşan herhangi bir yanlışlığa karşı yazılımın bakımı.

Yazılım ihtiyacını karşılama hedefine ilerlerken izlenecek farklı yollar, yazılım süreç modelleri olarak standartlaştırılmıştır. Süreç modeli aşamalarının sırasını ve aralarındaki geçiş kriterlerini belirler. Bir süreç modeli seçerek takip etmek, gerekli aşamaları hatırlatmaya ve planda önemli işlemlerin atlanmasına engel olmaya yarar. Bu modellerden bazıları aşağıdaki şekildedir (Nizam, 2015):

1. Kod eksenli (Kodla ve Düzelt)
2. Doğrusal Modeller
    a. Şelale (Waterfall) Modeli
    b. V Model
3. Evrimsel Süreç Modelleri
    a. Artımsal (Incremantal) Model
    b. Sarmal (Spiral) Modeller
    c. Rasyonel Bütünleştirme Süreci (RUP - Rational Unified Process)
4. Çevik (Agile) Metodolojiler
    a. Özelliğe Dayalı Geliştirme Modeli (FDD - Feature-Driven Development)
    b. XP (Extreme Programming)
    c. SCRUM

3.4.1. Kod Eksenli Yazılım Geliştirme

Kodla ve Düzelt yaklaşımı, kullanıcının isteğini hemen gerçekleştirmeye yöneliktir. Bu yaklaşım kullanılan projede ihtiyaç analizi bitmeden plansız şekilde kodlamaya başlanır veya ihtiyaç analizinin bir kısmı tamamlandıktan sonra tasarım yapmadan kodlamaya başlanır. Sistem mimarisinin baştan düzgün olmaması, sonraki aşamalarda sürekli değişim ve sorunlara sebep olur. İhtiyaç analizi tam yapılmadığından, müşterinin sürekli değişen isteklerine kodları değiştirerek çözüm yolu arayan yazılımcılar bir süre sonra kısır döngü içine düşerler. Kodun devamlı değiştirilmesi yazılımın kalitesini düşürür ve maliyetin artmasına sebep olur. Kodla ve Düzelt yaklaşımı Şekil 11'deki gibi olmaktadır:



Şekil 11. Kodla ve Düzelt Yaklaşımı

Kaynak: Nizam, (2015)

Kod eksenli yazılım geliştirmede eğitim ihtiyacı yoktur ve müşterinin beklentisine uygun olarak hemen üretime başlanmış izlenimi verir. Eğer ortaya çıkacak ürünü onu yapanlar tarafından kullanıcaksa avantajlıdır. Onun dışında avantajı yoktur, birkaç kişilik küçük projelerde kullanılır. Bireysel geliştiriciler için uygundur.

3.4.2. Doğrusal Modeller

Doğrusal modeller, yazılım sürecinde analizden canlıya alınıncaya kadarki aşamaların birbirini takip ettiği ve fazla geri dönüş yaşanmadığı modellerdir. Yazılım geliştirmede en sık kullanılan modeldir. Fakat günümüzde müşterinin istekleri ve teknik altyapıların sürekli değişmesi yüzünden bu modeller dezavantajlara sahiptir.

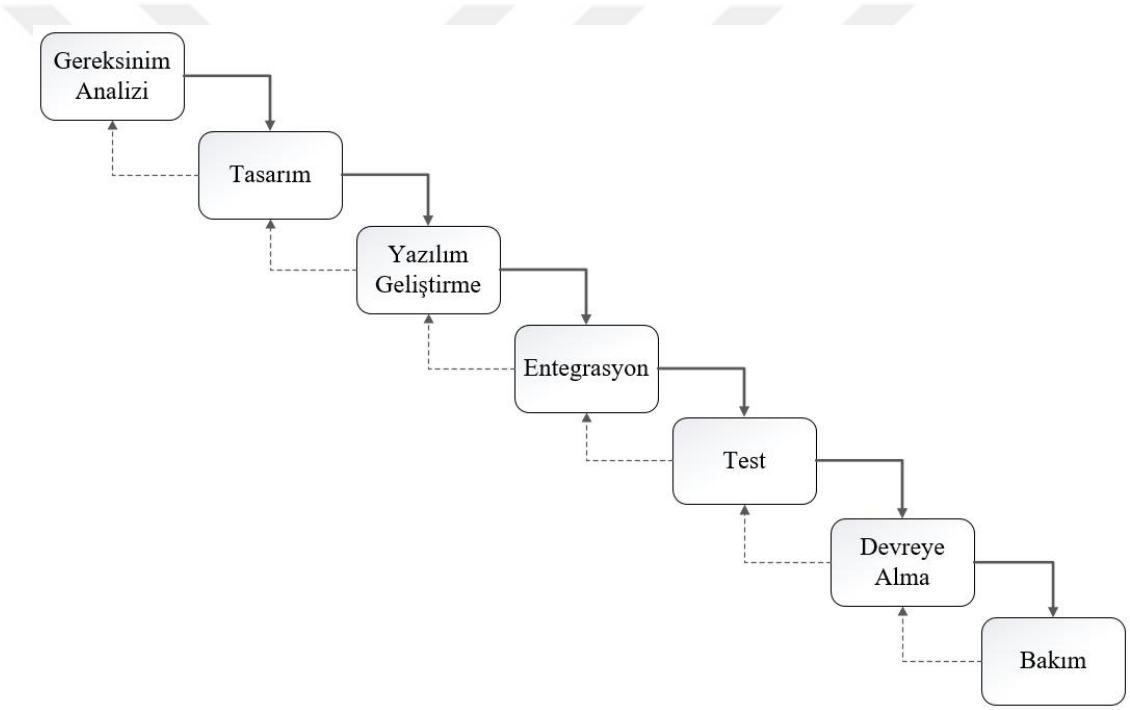
3.4.2.1. Şelale (Waterfall) Model

Şelale modeli şirketlerde yaygın olarak kullanılan ve ilk önerilen modellerdendir. Bu model 1956 yılında Benington tarafından Amerikan Donanması'nın düzenlediği bir matematik sempozyumunda sunulmuştur. 1983 yılında ilk model olarak Benington tarafından literatüre geçmiştir (Nizam, 2015).

Şelale modelinde proje analizden devreye alma aşamasına kadar doğrusal bir şekilde ilerler. Analiz aşamasında ihtiyaçların analizi yapılır, gözden geçirilir, kullanıcıdan onay alınır ve tasarım aşamasına geçilir. Tasarım doğru bir şekilde

tamamlanır, gözden geçirilir ve onay alınarak test aşamasına geçilir. Bir sonraki aşama bir önceki aşama bitmeden başlamamaktadır. Şelale modeli şekilde görüldüğü gibi 7 süreçten oluşmaktadır:

- Gereksinim analizi
- Tasarım
- Yazılım Geliştirme
- Entegrasyon
- Test
- Devreye alma
- Bakım



Şekil 12. Yazılım Geliştirme Şelale Modeli

Kaynak: Nizam, (2015)

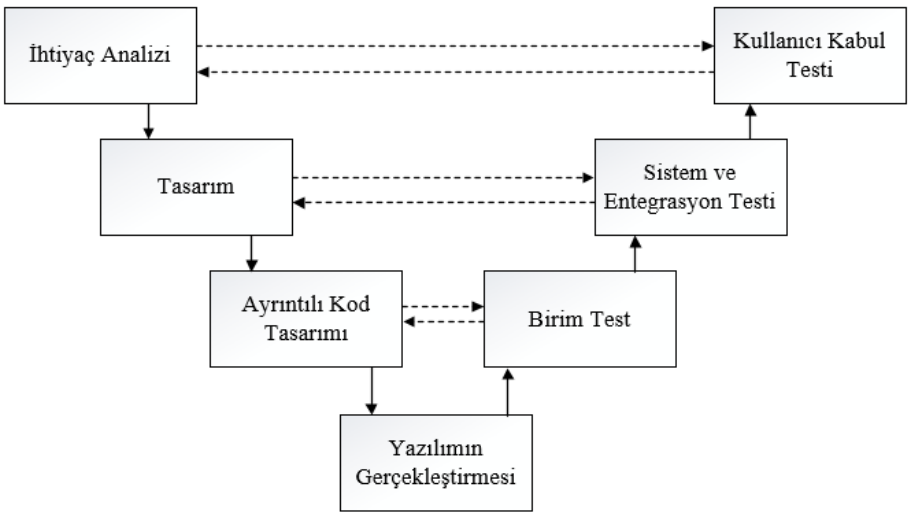
Şelale modeli Şekil 12'de görüldüğü gibi geri beslemeleri de olan bir geliştirme sürecidir. Bu modelde her aşamanın sonunda kontrol için gözden geçirme planlanır. Aşamaların sağlıklı bir şekilde yürümesi için ihtiyaç analizinin eksik olmaması ve değişkenlik göstermemesi gerekir. Bu modelle en büyük sorun kullanıcı ihtiyaçlarının ilk anda tamamen analiz edilememesinden

kaynaklanmaktadır. İstekler tam anlaşılsa da sonradan değiştirilebilir. Bu da bir çok projede bu modelin kullanımını zorlaştırır.

3.4.2.2. V Model

V Model, şelale modelinin kontrol safhalarının daha geliştirilmiş halidir. Her aşama kendi kontrol aşamasıyla eşleştirildiğinden "V" harfine benzer Şekil 13'te görülmektedir ve ismini buradan alır. V modelin kullanımı kolaydır. Bu modelde hangi aşamada nasıl test edileceği bellidir. Bu nedenle test ve doğrulama planlı bir şekilde ilerler. Hatalar bu yöntemle önceden fark edilir ve diğer aşamaya geçilmeden düzeltme sağlanabilir. Başarı oranı Şelale modele göre daha fazladır. Bu modelin, Şelale modelinin test kısmı standartlaştırılmış bir türü olduğunu ve yeni bir şey getirmediği ileri sürülmektedir (Şeker, 2015). İhtiyaçların yeteri kadar anlaşılamaması ve bu sebeple maliyetli geri dönüşlerin olması V modelinde de muhtemeldir.

V modelde, şartların açıkça tanımlandığı ve sabitlendiği küçük veya orta büyüklükteki projeler için kullanılmalıdır. E-Devlet uygulamaları, Askeri ve Savunma Sanayi yazılım projeleri, finansal yazılımlar gibi alanlarda bu model kullanılır.

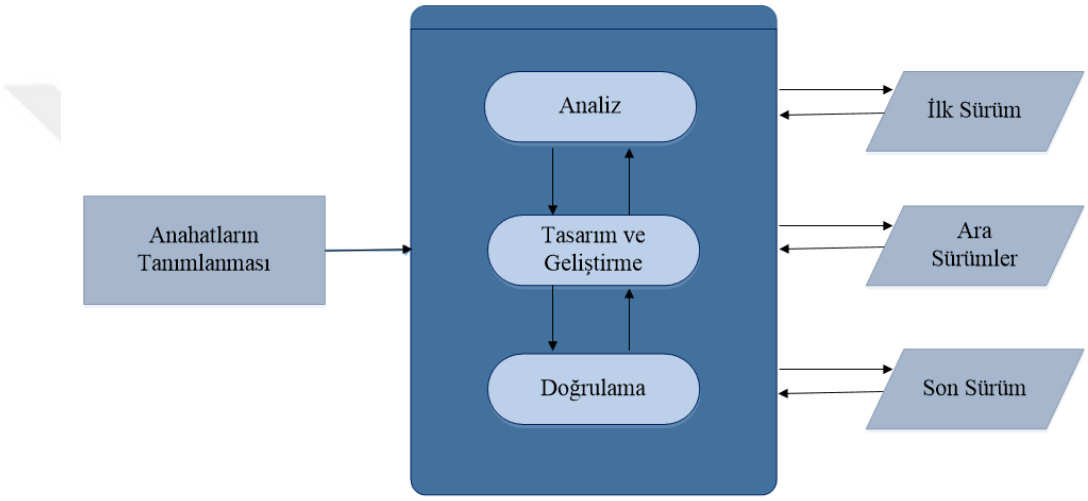


Şekil 13. V Model

Kaynak: Nizam, (2015)

3.4.3. Evrimsel Süreç Modelleri

Evrimsel süreç modelinde tanımlama, tasarım, analiz, geliştirme ve doğrulama faaliyetlerinin öncelikle hızlı bir şekilde yapılması, doğrulama sağlanana kadar aşamaların tekrarlanmasını temel alan bir modeldir (Nizam, 2015). Süreç aşamaları Şekil 14'teki gibi olmaktadır. Evrimsel süreç modelleri genellikle çok birimli organizasyonlar için önerilmektedir. Çok birimli banka uygulamaları gibi.

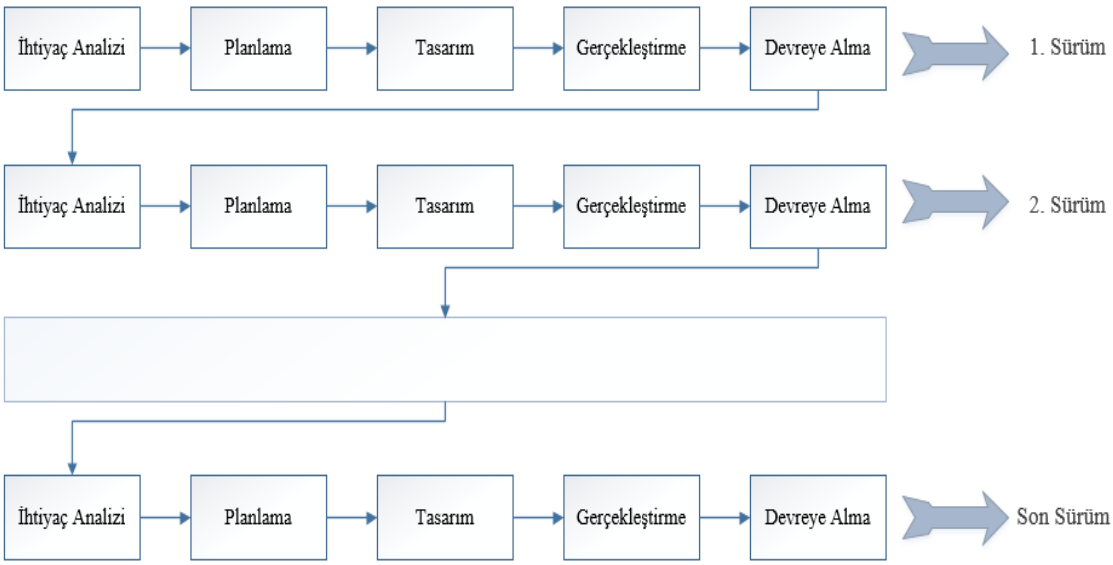


Şekil 14. Evrimsel Süreç Modeli

Kaynak: Nizam, (2015)

Evrimsel geliştirme üç farklı model öne çıkarmaktadır:

1. Artımsal (Incremantal) Model
2. Spiral Modeller
3. Rasyonel Bütünleştirme Süreci (RUP - Rational Unified Process)



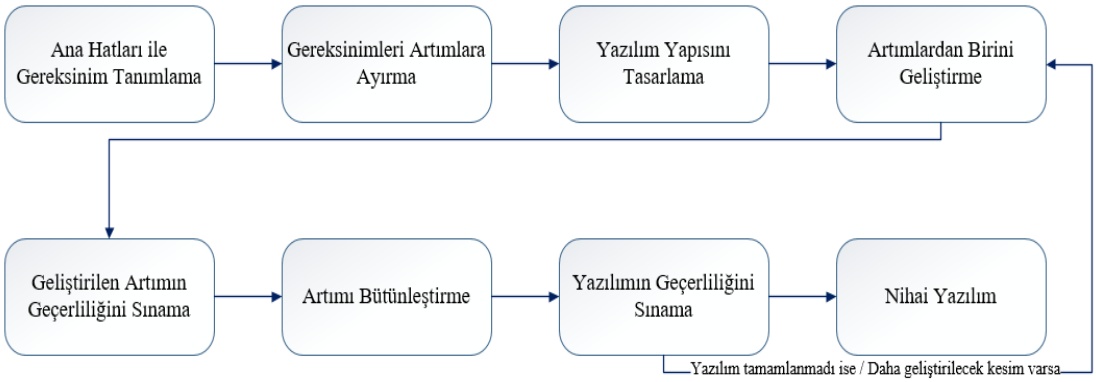
Şekil 15. Evrimsel Yazılım Geliştirme Modeli

Kaynak: Nizam, (2015)

Şekil 15'te görüldüğü gibi evrimsel geliştirme modeli, projenin ihtiyaç analizinden başlar, tasarım, geliştirme, test aşamalarını kapsayarak çevrimler şeklinde büyüyerek geliştirilmesidir. İhtiyaçlar büyük ölçüde belirlenmeden ilk sürüme başlanılmamalıdır. İhtiyaçlarda küçük çaplı değişiklikler projenin akışında sorun yaratmamaktadır. Evrimsel geliştirme modelinde her sürümde tekrar ihtiyaç analizi yapılır. Bu modeldeki çevrimlerin en fazla bir ay gibi kısa sürede tanımlanması gerekir.

3.4.3.1. Artımsal (Incremantal) Model

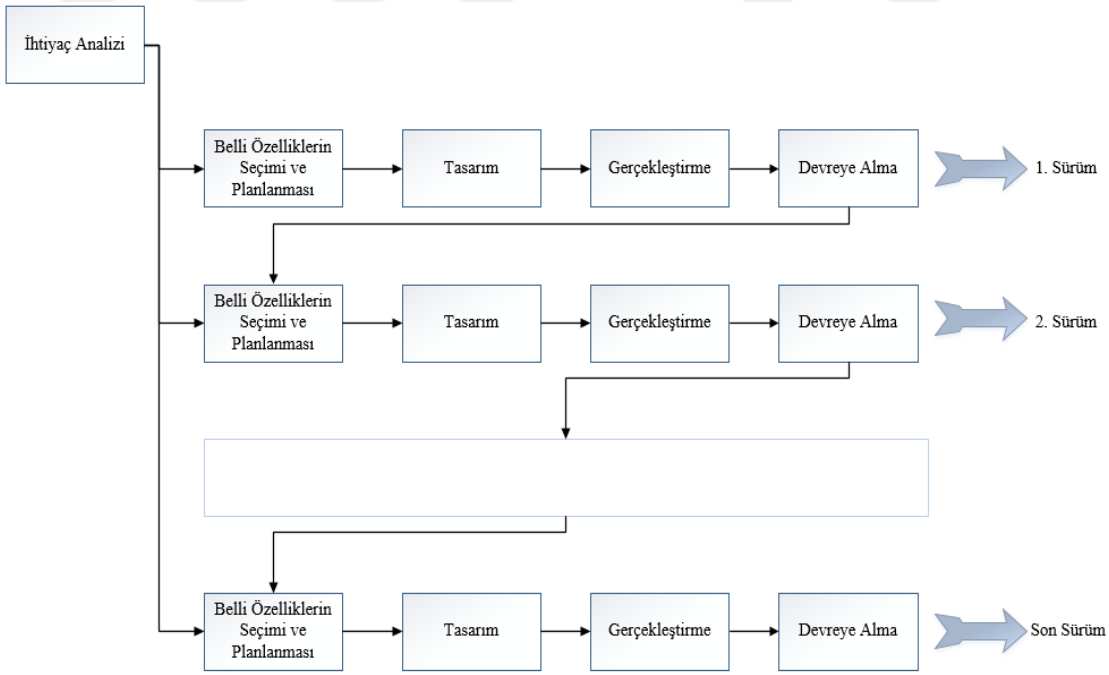
Artımsal model, Şelale modelindeki eksikliklerden yola çıkılarak geliştirilmiştir. Yazılımın geliştirilmesi sürecinde tekrar ile yazılımın daha iyi hale getirilmesi hedeflenmiştir. Şekil 16'daki süreci izleyerek yazılım kalitesini arttıran, erken geribildirim sayesinde riskin minimuma indirgenmesini sağlayan, yeni ve değişken gereksinimleri gerçekleştirmek amacıyla esnek bir yapı sunan bir metottur (Nizam, 2015).



Şekil 16. Artımsal Model

Kaynak: Nizam, (2015)

Artımlı geliştirme modelinde, ihtiyaç analizinin büyük bir kısmı belli olduktan sonra yazılım geliştirme süreci sürümlere bölünerek ve her bir sürüm tamamlandıkça ihtiyaçların bir kısmı karşılanarak gerçekleştirilmesidir. Nizam (2015)'a göre Artımlı geliştirme modeli Şekil 17'deki gibidir:



Şekil 17. Artımlı Geliştirme Modeli

Kaynak: Nizam, (2015)

Sistemin çalışan yeni sürümü birkaç haftada bir ortaya çıkarılır. Sürümler geliştirilir, test edilir, proje ekibinden veya kullanıcı onayı alınır. Sürümlerin özellikleri hedef kitle göz önününe alınarak belirlenmelidir. Çünkü bazı sürümler proje ekibini hedef alırken, bazı sürümler üst yönetimi, bazı sürümler ise direk son kullanıcının ihtiyaçlarını ayrıntılı olarak geliştirmeyi hedef alır. Bu modelde hatalar kısa sürede fark edilir ve sürecin sürekli izlenmesi sağlanır.

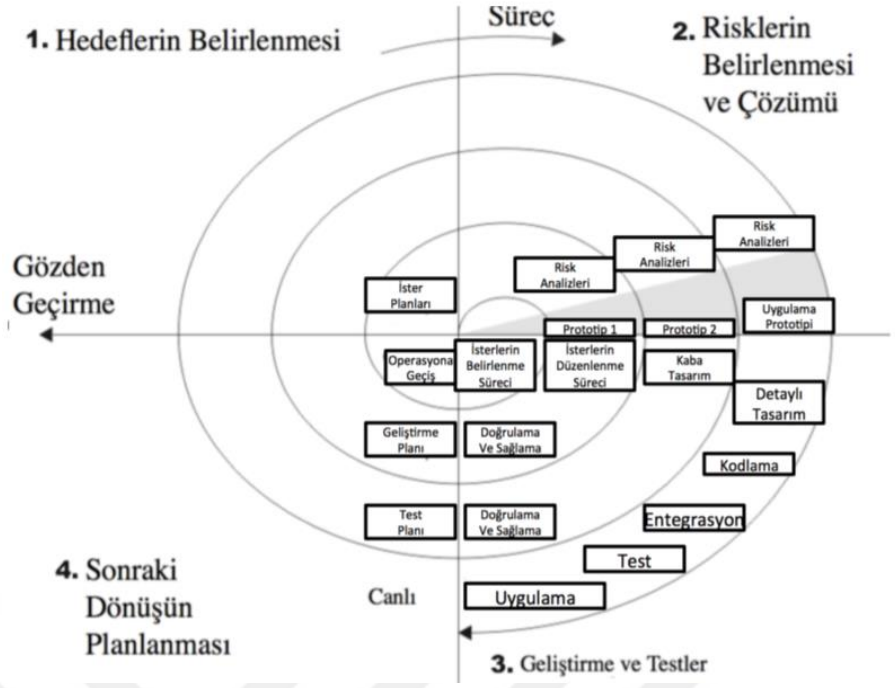
Artımlı geliştirme modelinin başarılı olması projenin ilk aşamasında ihtiyaçların büyük ölçüde belli olmasına bağlıdır. Analizi sıklıkla değiştirilen projelerde bu model başarılı olmamaktadır. Her sürümde bir önceki sürüme bağlı olarak planlar geliştirilmelidir.

3.4.3.2. Sarmal (Spiral) Modeller

Sarmal (Spiral) model ilk olarak Barry Boehm tarafından 1988 yılında A Spiral Modeli ile Yazılım Geliştirme ve Geliştirilmesi adlı makalesinde tanımlanmıştır (Şeker, 2015). Bu modelin iki temel avantajı bulunduğu kabul edilir. Bunlar:

- Sistemin anlaşılır ve gerçekleştirilen kısmı büyürken, riski azalır.
- Tanımlanan hedefleri vasıtasıyla geliştirilen sistemin ihtiyaçlarıyla uygunluğu konusunda kullanıcılarla mutabık kalınması ve müşteri memnuniyetinin arttırılması gereklidir.

Modelde her aşama tasarım modeli ile başlar, gözden geçirme ile devam eder, müşteri ile sona erer. Analiz ve yazılım çalışmaları modelin her aşamasında bulunmaktadır. Spiral modelde aşamalar üzerinden tekrar tekrar geçilir.



Şekil 18. Spiral Model

Kaynak: Şeker, (2015)

Şekil 18'de görüldüğü gibi 4 bölüme ayrılır. 4 bölüme ayrılmasının nedeni projeyi küçük parçalara ayırarak riskin minimize edilmesi ve bu sayede geliştirme süresinde kolay değiştirilebilir olmasının sağlanmasıdır. Spiral şeklinde dönerek aşamalar ilerletilir. Her döngüye paydaşların tespiti ile başlanır. Dönme esnasında ürünün maket uygulaması olan prototipler çıkarılır. Prototipler müşteriye gösterilir, görüşleri alınır, istek ve ihtiyaçları belirlenir. Risk analizleri yapılıp tekrar prototipler çıkarılır. Bu prototipler doğrulama ve onaylama aşamasından geçirildikten sonra tekrar yazılım sürecinden geçer. Tekrar risk analizi yapılır ve prototip 2 çıkarılır. Bu aşamada test planı hazırlanır ve risk analizi yapılarak uygulamaya geçmeden önceki son prototip çıkarılır. Bu aşamada projenin büyüklüğüne göre prototiplerin sayısı artırılabilir. Uygulama prototipinden sonra tasarım, kodlama, entegrasyon, test ve devreye alma süreci başlar. Süreçlerin sonunda model şelale modeline benzemektedir. Her prototip sonunda yapılan müşteri değerlendirmelerine göre memnuniyet sağlanana kadar devam eder. Çünkü her iterasyonda belirlenen istek ve ihtiyaçlara göre yeniden yapılandırma olur ve tekrar tekrar test sürecinden geçer.

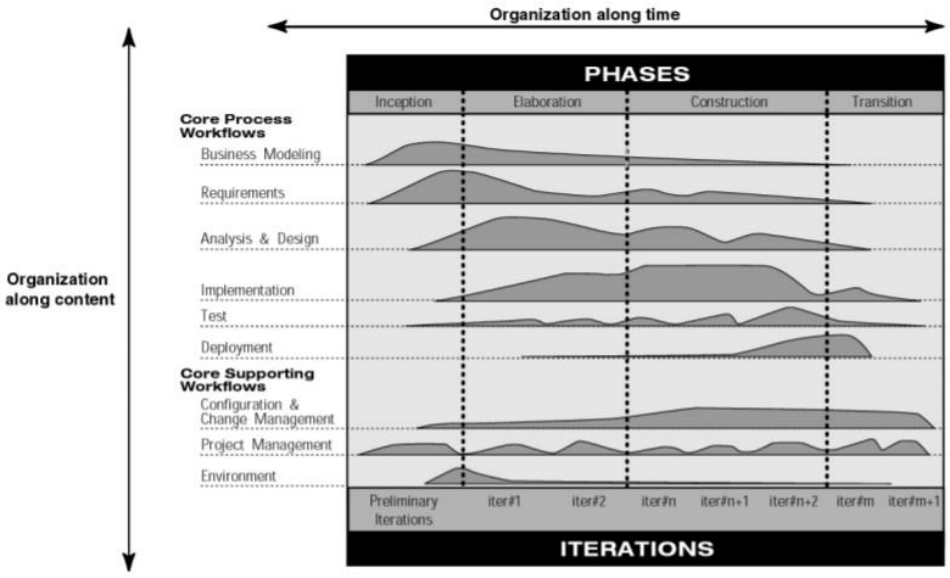
- Orta dereceli yüksek riskli projelerde,
- Ekonomik potansiyel önceliği bulunan uzun vadeli projelerde,

- Kullanıcılar istek ve ihtiyaçlarından emin değillerse,
- Gereksinimler karmaşıksa,
- Yeni ürün hattı oluşturulacaksa,
- Önemli değişiklikler (araştırma ve inceleme) bekleniyor ise spiral model kullanılması en uygun model olacaktır.

Özetle spiral model, farklı döngülerden geçerek risk analizlerinin sıkça yapıldığı, yazılım ekibinin, proje yönetim ekibinin ve müşterinin onayladığı bir prototipin ortaya çıkarıldığı ve sonrasında bu prototipin gerçekleştirilip devreye alındığı bir modeldir denilebilir.

3.4.3.3. Rasyonel Bütünleştirme Süreci (RUP - Rational Unified Process)

Rasyonel Bütünleştirme süreci; yinelemeli, artımsal yazılım geliştirme süreci modelidir (Karadağ, 2002). Bu modelde ihtiyaçların neler olduğu, çalışanların sorumlulukları belirlidir. Rasyonel Bütünleştirme Süreci, çapraz fonksiyonel projelerde iyi çalışır. Rasyonel Bütünleştirme Süreci, Şekil 19'da görüldüğü gibi bir süreç üst yapısıdır, geleneksel biçimde kullanılabilir (Karadağ, 2002).

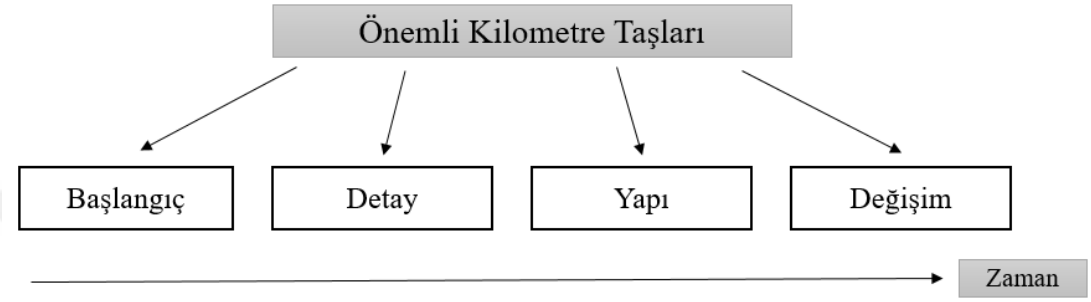


Şekil 19. Rasyonel Bütünleştirme Süreci Genel Mimarisi

Kaynak: Karadağ, (2002)

RUP sürecinin Şekil 20'de görüldüğü gibi iki boyutu vardır:

1. Yatay eksen zaman göstergesidir ve sürecin yaşam döngüsü yönlerini gösterir.
2. Dikey eksen iş akışlarını göstermektedir. İlk boyut sürecin dinamik yönünü, fazları, iterasyonları ve kilometre taşları ile ifade edilir. İkinci boyutta sürecin statik yönü belirtilir.



Şekil 20. RUP Fazları

Kaynak: Nizam, (2015)

RUP sürecinde şekildeki gibi yaşam döngüsü dört faza ayrılmıştır. Her fazda, fazın amaçlarının yerine getirilip getirilmediği değerlendirilmesi yapılır ve değerlendirme sonrası projenin bir sonraki aşamasına geçilir.

3.4.4. Çevik (Agile) Metodolojiler

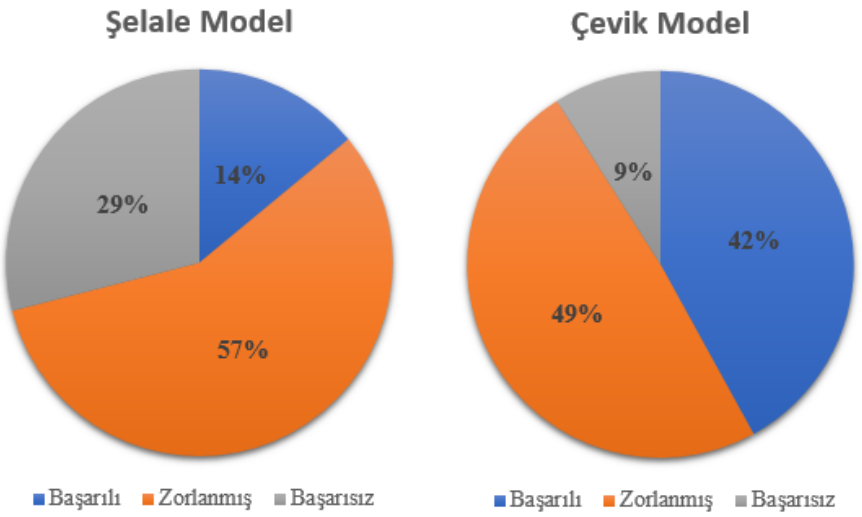
Türk Dil Kurumu'na göre çevik kelime olarak kolaylık ve çabuklukla davranan, tetik, atik demektir. Bilişim'de ise çevik kelimesi değişen ihtiyaçlara hızlı cevap veren pratikleri ifade etmektedir. Yazılım süreçlerini kısaltan bir yazılım geliştirme metadolojisidir.

1950'li yıllarda üretim alanındaki verimliliği arttırmak amacıyla geliştirilen yalın yaklaşımların yazılım sektöründeki bir uzantısı olarak ortaya çıkmıştır. Bilişim alanına 1970'li yıllardan itibaren ortaya çıkmış, 1990'larda hızla kullanılmaya başlanmış ve geçtiğimiz son 10 yılda tüm dünyada başarısını kanıtlayarak popülaritesini arttırmıştır. Hızla yayılmasının nedeni çevik yaklaşımın BT projelerine yapılan yatırımların sonuçlarını daha hızlı görülmesini sağladığından ve müşteriye

katma değer yaratmaya odaklı olmasından kaynaklanmaktadır. Çevik metotlar, piyasaya hızlı bir şekilde ürün sunabilme, değişen isteklere karşı hızlı yanıt verebilme ve minimum sürede yazılım ürününü müşteriye sunabilmeyi amaçlamaktadır (Boehm ve Turner, 2003).

Çevik yazılım geliştirmenin en önemli prensibi kaliteli yazılım ve hızlı üretim ile müşteri memnuniyetini sağlamaktır. Projenin her sürecinde değişiklik yapılabilir ve bu durum kullanıcı avantajına dönüştürülebilir. Kısa zaman aralıklarında çalışılır ve kaliteli yazılım üretilir. Kendini organize edebilen sorumluluk sahibi ekip çalışanları günlük olarak yüz yüze iletişim halinde çalışır. Yazılımda basitlik hedeflenir ve öncelikli gelişim ölçütü çalışan yazılımdır.

Çevik geliştirme metotları değişim odaklı olduğundan küçük ve orta çaplı uygulamalar için uygundur. Sadece müşteri isteklerini dikkate almak, yazılım teknolojisinin gelişimi açısından olumsuz sonuçlara neden olur. Şekil 21'de Standish Group'un yapmış olduğu araştırmada Çevik metotların başarı oranının Şelale metoduna göre daha yüksek olduğu görülmektedir.



Şekil 21. Yazılım Geliştirme Modelleri Başarı Oranları

Kaynak: Hastie, (2015)

Tablo 9'a göre Projelerde Çevik model Şelale modeline göre daha başarılır.

Tablo 9. Standish Group Raporuna Göre Çevik-Şelale Başarı Oranları

Projelerin Büyüklüğü	Yöntem	Başarılı	Sorunlu	Başarısız
Tüm Projeler	Çevik	39%	52%	9%
	Şelale	11%	60%	29%
Büyük Projeler	Çevik	18%	59%	23%
	Şelale	3%	55%	42%
Orta Büyüklükte Projeler	Çevik	27%	62%	11%
	Şelale	7%	68%	25%
Küçük Projeler	Çevik	58%	38%	4%
	Şelale	44%	45%	11%

Kaynak: Hastie, (2015)

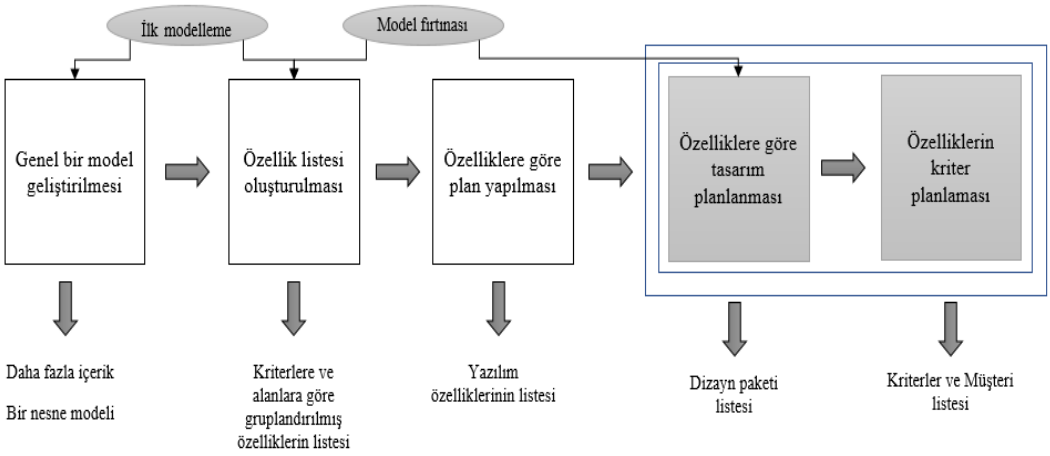
3.4.4.1. Özelliğe Dayalı Geliştirme Modeli (FDD - Feature-Driven Development)

Özelliğe dayalı geliştirme modeli, yazılım geliştirme yaşam döngüsünde, tasarım ve geliştime aşamalarını etkileyen yazılım geliştirme yöntemidir. Beş ana aktivitesi bulunmaktadır (Palmer ve Felsing, 2001);

1. Tüm sistemleri modelleme (Ekip üyeleri sistem için yol planı hazırlar.)
2. Özellik listesi hazırlama (Gerekli özellikler tüm taraflar anlayacağı şekilde parçalara bölünür.)
3. Özellikleri planlama (Tasarım paketleri adı verilen bu parçalar önceliklendirilir ve ilgili yazılım sorumlusuna atanır.)
4. Özelliğe göre tasarım (Sürecin iteratif kısmı başlar. Yazılım sorumlusu kendine atanan parçalardan sıralama yapar.)
5. Özelliğe göre geliştirme (İlk seçilen parça için analiz, test ve entegrasyon süreci başlar).

Tüm süreçlerde kalite unsuru ön planda tutulur. Kod yazıldıktan sonra birim test ve kod inceleme alt süreçlerine geçilir. Hangisinin yapılacağına yazılım sorumlusu karar verir. Kod inceleme FDD için zorunludur ve bu süreç için kod yazma süresinin dörtte biri kadar süre verilir. Kod inceleme düzgün yapıldığında test aşamasına geçilmeden yazılım kusurlarının çoğu bulunur. Bu süreç yazılımcılarının daha dikkatli ve sağlıklı program yazmalarını sağlar. Kodlardaki etki ve karmaşıklık

yazılım sorumlusu tarafından tespit edilir. Karmaşık kod incelemelerinde diğer takımlardaki yazılım sorumlulukları da davet edilir. XP programlamada yapılan çift programlama tekniğinden daha faydalıdır. Bu süreçte yazılımcını koda farklı bir gözle bakması sağlanır ve kodun son hali entegrasyona hazır hale gelir (Harleen ve Swati, 2014). Özelliğe dayalı geliştirme modeli Şekil 22'deki gibidir:



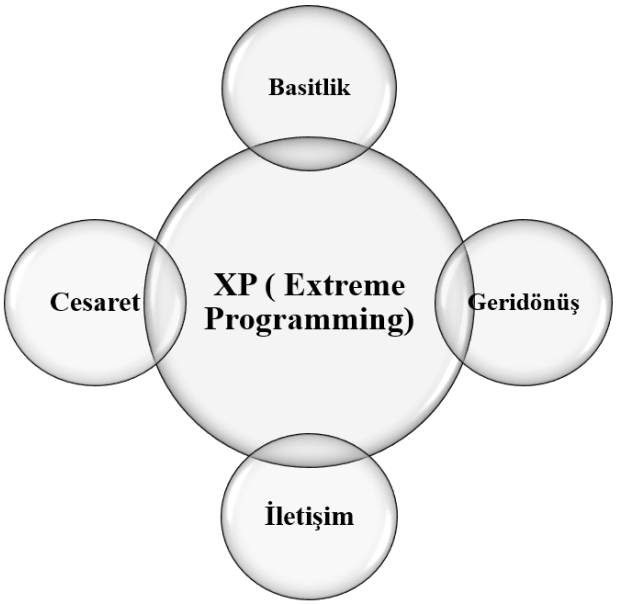
Şekil 22. Özelliğe Dayalı Geliştirme Modeli

Kaynak: Palmer ve Felsing, (2001)

3.4.4.2. XP (Extreme Programming)

XP, Kent Beck aracılığıyla 1996 yılında Chrysler firmasında yapılan bir projede oluşturulmuştur. XP metodolojisinde müşteri istek ve gereksinimleri merkezi rol oynamaktadır. XP ile yazılımda, devamlı değişen müşteri gereksinimlerine hızlı bir şekilde uyum sağlar. Projenin başında müşteri gereksinimleri detaylı bir şekilde dokümante edilir. Oluşturulan bu dokumanlar baz alınarak yazılıma başlanır. Süreçler ilerledikçe müşteri tarafından yapılması istenilen değişikliklerin maliyetleri oldukça yüksek olur, çünkü mevcut yapı yani tasarım istenilen değişikliklerin yapılmasına engel oluşturabilir ve bu yüden yeni bir yapılandırılmaya gidilmesi gerekebilir. Projede çevikliğin sağlanabilmesi için az yükle başlamak gerekir. XP'nin kullanıldığı projelerde formalite ve bürokrasi ez az seviyeye çekilir. Bu yüzden projenin başında yani yazılıma daha başlanılmadan geniş çapta tasarım ve son detaylı gereksinim dokümanı oluşturulmamaktadır (Acar, 2008).

XP; Basitlik, Geridönüş, İletişim ve Cesaret olmak üzere dört değeri baz alır. Bu değerlerin tamamı uygulandığı taktirde XP kullanımı kolaylaşır. Verimli bir çevik süreç oluşturulması için Şekil 23'te görülen bu dört değerin kabul görmesi gereklidir.



Şekil 23. XP (Extreme Programming) Değerleri

Kaynak: Acar, 2008

XP'de basit yöntemlerle sonuca ulaşılır. Bu şekilde hızlı ve düşük maliyetlerle proje gerçekleştirilir. Basit çözümlerle oluşturulan programın bakımı ve geliştirilmesi kolay olmaktadır. Basit çözümler kolay öğrenileceğinden zaman kazancı da sağlanmaktadır. XP projelerinde kalite kontrol geri dönüşüm (feedback) üzerinden sağlanır. Yazılımcılar programlarının kalitesini testlerin geri dönüşümünden ölçerler. Geri dönüşümler sayesinde program üzerinde entegrasyonlar yapılarak son hali müşteriye sunulur. XP'nin oluşabilmesi için her süreçte geri dönüşüm mekanizmalarının oluşturulması gereklidir.

Projelerde ekip üyelerinin sürekli iletişim halinde olmaları gereklidir. XP'de çalışanların yüz yüze görüşmeleri büyük önem taşımaktadır. Bu sayede sağlıklı bilgi akışı sağlanır ve yanlış anlaşılmalar ortadan kaldırılır. Takım içerisindeki iletişim bağının kuvvetli olması zaman kazanılmasını da sağlar ve projenin ilerleme hızını da büyük ölçüde etkiler. Basitlik, geridönüşüm ve iletişim için cesaret gereklidir.

Cesaret ekibin birey bazında iç dünyalarını terk ederek, takımın bir parçası olmalarını kolaylaştırır. XP metodolojinde Müşteri (Customer), Programcı, Proje Menajeri, Koç ve Testçi rolleri bulunmaktadır (Acar, 2008).

- ***Müşteri,*** istek ve ihtiyaçlarına cevap verebilecek yazılım için yatırım yapan kişidir. Müşteri proje başında isteklerini kullanıcı hikayeleri oluşturarak ifade eder.
- ***Programcı***, müşterinin istek ve ihtiyaçlarına uygun programın yazılımın tasarım, kodlamasını ve testini yapan kişidir. Takımla birlikte test güdümlü olarak çalışır. XP'de programcılar kod yazımına başlamadan önce test sınıflarını oluşturarak implementasyona yani entegre etmeye başlarlar.
- ***Proje Menajeri***, müşteri ve programcıları biraraya getiren kişidir. Proje menajeri görev dağılımını yapmamaktadır, görevi ekibin kendi başına sorumluluk almalarını sağlamaktır. Toplantıları koordine eder ve takımın karşılaştığı sıkıntılara çözüm yolu arar.
- ***Koç***, çevik süreci ve uygulamasını bilen kişidir. Görevi proje başında çevik takımı oluşturarak onlara bu süreçte rehberlik etmektir. Projede sıkıntı oluştuğunda ve takımın XP dışına çıktığı zamanlarda müdahale eder.
- ***Testçi***, programın testlerini implemente eden programcıdır.

XP'de süreç ilerleyişi aşağıdaki şekilde olmaktadır:

- Müşteri isteklerini kullanıcı hikayeleri olarak oluşturarak öncelik sırası atar. Programcılar bu kullanıcı hikayeleriyle ilgili zaman tahminlemesi yaparlar.
- Müşteri programcılarla birlikte iterasyon ve sürüm planını hazırlar. İterasyon genellikle 1-2 hafta ve sürüm ise 1-2 ay olmaktadır.
- Programcılar kullanıcı hikayelerinin üzerinden geçer ve sorularla ilgili müşteriye danışırlar.
- İlk süreçten sonra programcılar müşteriye çalışır durumda program sunarlar ve müşteriden geridönüşüm sağlarlar.
- Alınan geridönüşümlerden sonra ikinci süreç planlanır. Müşteri yeni gereksinimlerini belirtirse, bunlar için tekrar kullanıcı hikayeleri oluşturulur. Yeni bi gereksinim yoksa mevcut kullanıcı hikayelerinden öncelik sırası yüksek olandan devam edilir.

- İlk sürüm sonunda oluşan program müşteri tarafından kullanıma alınır ve başka sürümler planlandıysa bir sonraki iterasyon ile devam edilir.

XP dört safhadan oluşmaktadır. Bunlar (Acar, 2008):

- **Keşif Safhası (Exploration Phase):** Projenin başında müşterinin kullanıcı hikayeleri ve programcının teknik alt yapı araştırmaları keşif safhasını oluşturur.
- **Planlama Safhası (Planning Phase):** Bu safhada müşteri ve programcılar beraber iterasyon ve sürüm planlarını oluştururlar. Zaman tahminlemesi yapılır. Kullanıcı hikayeleri önceliklendirilir. İş bölümlendirmesi yapılır.
- **İterasyon ve Sürüm Safhası (Iterations to elease Phase):** Kullanıcı hikayelerinin implemantasyonu bu safhada gerçekleştirilir. Hangi kullanıcı hikayelerinin implemente edilmesine müşteri karar verir. Her iterasyon sonunda müşteriye çalışır durumda program sunulur ve geridönüşümü sağlanır. Çalışma hızları ve kalite kontrol edilerek diğer iterasyon öncesi plan hazırlanır. İterasyonda ortaya çıkan hatalar bir sonraki süreçte giderilmek üzere planlanır.
- **Bakım Safhası (Maintenance Phase):** Bu safhada ortaya çıkan programın bakım ve geliştirilmesi sağlanır. Sistem hataları giderilir ve küçük çaplı eklemeler yapılabilir. Programla ilgili eğitimler bu safhada yer alır. Müşterilerin istekleri büyük ölçüde değiştiği takdirde ilk safhaya yani keşif safhasına dönülmesi gereklidir.

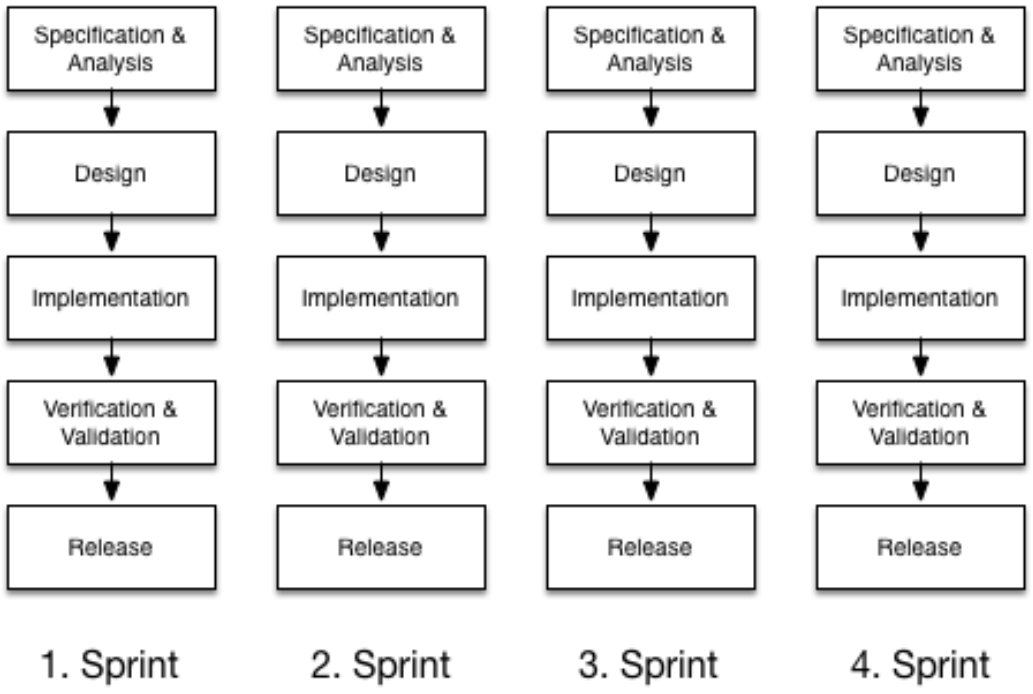
3.4.4.3. SCRUM

SCRUM; karmaşık problemlere çözüm sağlayarak, üretken ve yaratıcı bir şekilde olası en yüksek değerde ürünlerin üretilmesini sağlayan bir altyapıdır (Yitmen, 2015). SCRUM:

- Basittir,
- Anlaması kolaydır,
- Ustaca yönetmek zordur.

SCRUM, 1990'ların başında karmaşık ürün geliştirme sürecini yönetmek amacıyla kullanılan bir süreç çerçevesidir. SCRUM, ürün geliştirme tekniği ya da süreci değildir: içeriğinde çeşitli süreçleri ve teknikleri kullanabileceği bir çerçevedir (Yitmen, 2015).

SCRUM çevik modelinde, geliştirme süreci Şekil 24'teki gibi bir aydan kısa süren koşulara (sprint) bölünür (Sutherland ve Schwaber, 2016). SCRUM kullanılan projede şelale modelindeki gibi kullanıcı istekleri bir sprint içinde değiştirilemez. Sprint; SCRUM Geliştirme Takımı'nın çalışır durumdaki yazılım parçacığını geliştirmek için çalıştığı süredir.



Şekil 24. SCRUM Sprintleri

Kaynak: Yitmen, (2015)

SCRUM'da müşteri gereksinimleri Kullanıcı Hikayesi (User Story) olarak tanımlanır ve bu gereksinimler Özellikler Listesi (Backlog List)'nde tutulur. Böylece her sprint sonunda yazılımın bir parçası tamamlanır ardından tamamlanan her parça kullanıcıya sunulur. SCRUM'da analiz, yazılım, test için ayrı alt gruplar ve unvanlar bulunmamaktadır.

SCRUM'da proje yöneticisi rolü yoktur, SCRUM Master vardır. SCRUM Master'ın görevi, takımın SCRUM uygulayarak üretkenliğini arttırmasına destek olmaktır. Product Owner (Ürün Sahibi): Üretilecek olan ürünün değerini optimize etmekten sorumlu kişidir. Kullanıcı hikayelerini belirleyen kişidir. SCRUM Takımı:

- SCRUM Master,
- Product Owner (Ürün Sahibi),
- SCRUM Geliştirme Takımı (Development Team)'ndan oluşmaktadır.

SCRUM Geliştirme Takımı ise kendi kendine organize olarak Sprint'ler halinde çalışır durumdaki yazılım parçacıklarını üretmekten sorumlu olan kişilerdir. SCRUM, takımın kendi kendini yönetmesini amaçlayan bir metodolojidir. Yaratıcı fikirlerin ortaya çıkması, işlerin hızlanması ve daha motive çalışanlarla birlikte üretkenliğin arttırılabilmesi için takımın yaptığı işe sahip çıkması oldukça önemlidir. Bu sebeple yönetimsel sorumluluğun takım tarafından alınması ve takım içerisindeki her bir çalışanın karar mekanizmasının içerisinde aktif bir şekilde bulunması gerekmektedir (Yitmen, 2015).

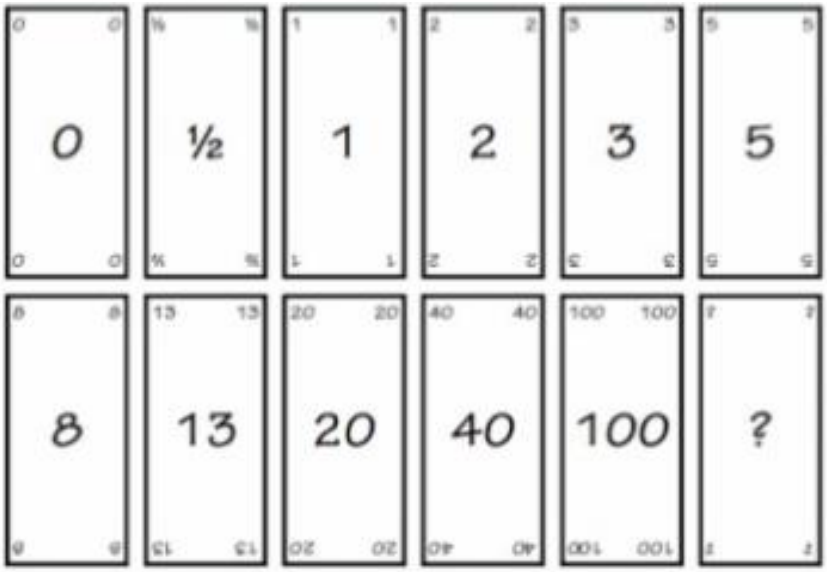
SCRUM riski kontrol edebilmek için artımlı (incremental) bir yaklaşım kullanır. SCRUM şeffaflık, gözlem ve adaptasyonla desteklenir.

- ***Şeffaflık***: tüm katılımcıların sürece dair dili ortak bir dil olmalıdır. SCRUM Takımı ve Ürün Sahibi arasında ortak bir 'Bitti' tanımına sahip olmalıdırlar. Bitti (Done): SCRUM Takımı'nın bir işin hangi koşul ve standartlarda tamamlanmış olarak sayılacağını tanımlanmasıdır.
- ***Gözlem***: SCRUM uygulayıcıları istenmeyen durumlarını belirleyebilmek için sıkça gözlem yapmalıdırlar. Gözlemler çalışma esnasında sorumluluk verilmiş gözlemciler tarafından yapılmalıdır.
- ***Adaptasyon***: SCRUM süreci olması gerekenin dışına çıktığında sürecin sonunda ürün kabul edilemez olacağı tespit edilirse üzerinde çalışılan süreç ve üründe iyileştirmeler veya geliştirmeler yapılmalıdır. Bu iyileştirme ve geliştirmeler çok uzun süreli olmamalıdır.

SCRUM'da rutin bir şekilde uygulanması gereken zorunlu dört toplantı süreci bulunmaktadır. Bunlar (Yitmen, 2015):

1. **Günlük SCRUM Toplantısı:** SCRUM Geliştirme Takımı'nın bir araya gelerek senkronize olduğu ve Sprint hedefine doğru ilerleyişini değerlendirdiği günlük planlama toplantısıdır.
2. **Sprint Planlama Toplantısı:** SCRUM Takımı'nın ilgili Sprint içerisinde yapılacak istekleri ve bu istekleri nasıl hayata geçireceğine dair stratejisini belirlediği toplantıdır.
3. **Sprint Review Toplantısı:** Sprint içerisinde üretilmiş olan çalışır durumdaki yazılımın değerlendirildiği toplantıdır.
4. **Sprint Retrospective Toplantısı:** Sprint sonunda ilgili Sprint'in değerlendirildiği ve iyileştirmeye yönelik adımların oluşturulduğu toplantıdır.

Bu toplantılar için Timebox belirlenir. Timebox; Bir işin yapılması için tanımlanmış belirli uzunluktaki zaman periyodudur. Bazen süre tahminleme de sıkıntılar oluşabilir. Bunun için ekip üyeleriyle Şekil 25'teki gibi planlama kartları ile oyun oynanır (Yitmen, 2015).

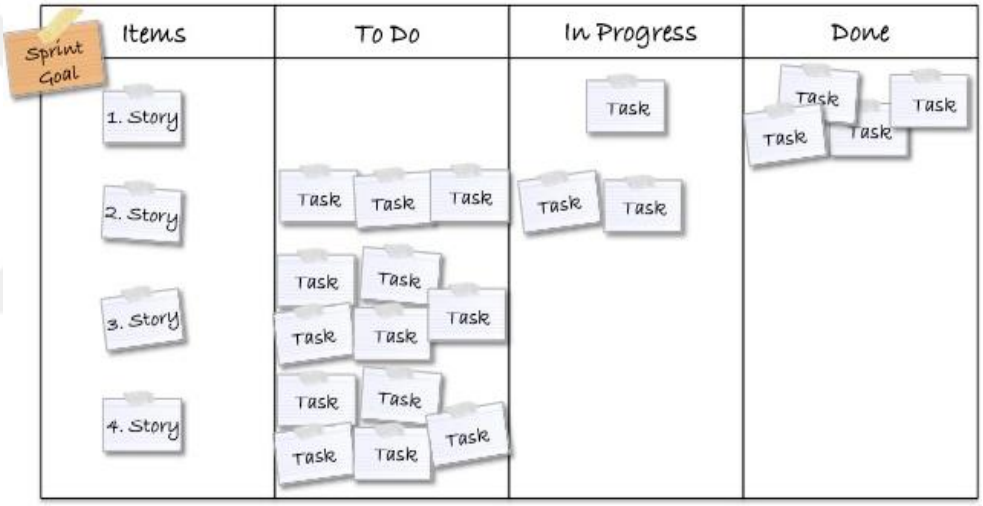


Şekil 25. Planlama Kartları

Kaynak: Yitmen, (2015)

Zaman tahmini yapılırken yazılımcının ideal şartlarda 8 saat program yazması öngörülmektedir. Fakat yazılımcı 8 saatin tamamını verimli olarak kullanamamaktadır. Sprint Planlama toplantısında seçilen gereksinimlerle Sprint Backlog oluşturulur. Kullanıcı hikayeleri gözden geçirilerek görev (task) listesi hazırlanır ve bu görevler görev kartlarına yazılır. Sprint içeriği ve durumunu devamlı takip edilebilmesi için Şekil 26'daki gibi SCRUM Tahtası kullanılmaktadır (Yitmen, 2015). Bu görev tahtasında;

1. Sütunda kullanıcı hikayeleri (Story),
2. Sütunda görevler (To Do),
3. Sütunda çalışma (In Progress),
4. Sütunda teslime hazır (Bitti) olan parçalar bulunur.



Şekil 26. SCRUM Tahtası

Kaynak: Yitmen, (2015)

4. DEĞER AKIŞ HARİTALAMA YÖNTEMİ

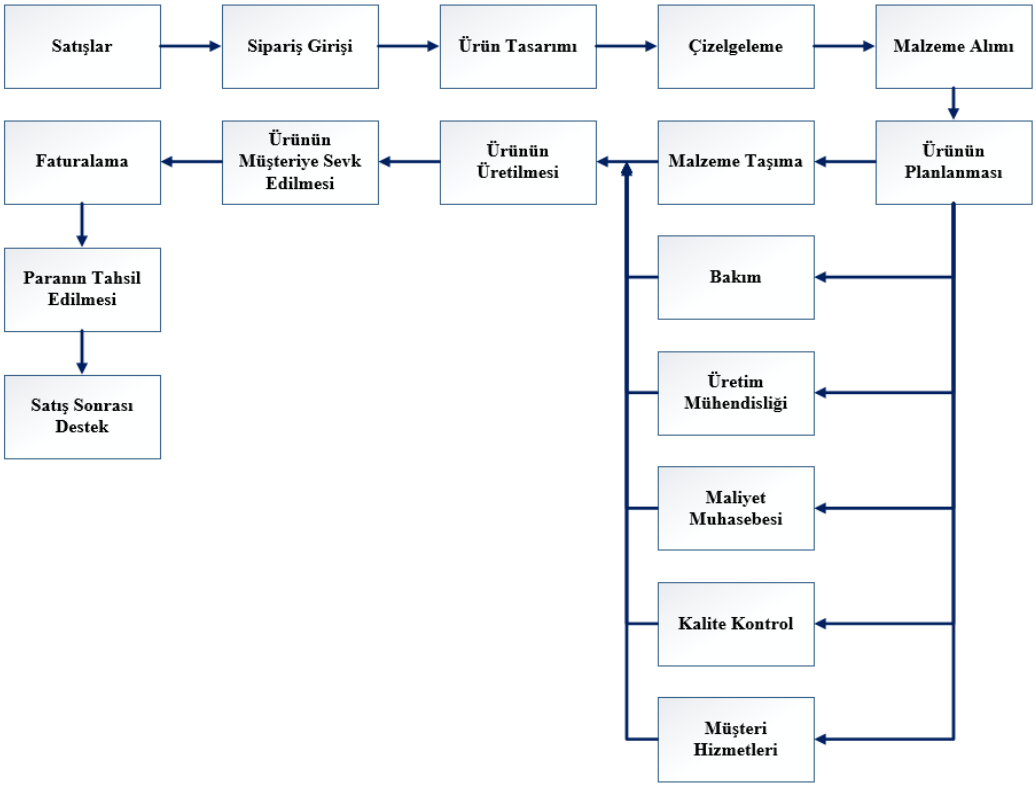
Son yıllarda firmalar kendi üretim alanlarında yalın ilkeleri uygulamaktadır. Sadece üretim alanına odaklanmak, iş süreçlerinde herhangi bir çaba harcamadan yalın bir firma haline gelmek yeterli değildir. Değer akış haritalama, yalın üretimi sağlamak için değerli bir araçtır. Firmalarda değer akış haritası uygulamak iş sistemlerine ve üretim sistemlerine yarar sağlamaktadır. Bu bölümde Değer Akış Haritalama ve Yalın Felsefe'den bahsedilecektir. Değer akışı, temel süreçler boyunca bir ürünü meydana getirmek için katma değer yaratan ve yaratmayan faaliyetler bütününe denilmektedir. (Rother ve Shook, 1999) Her bir ürün için geçerli olan ana akışlar aşağıdaki gibidir:

1. Hammaddeden müşteriye üretim akışı. Bu akışta genellikle yalın üretim ile ilişkilendirilir. Yalın tekniklerin uygulandığı alandır.
2. Ürünün geliştirme süreci.

Yalın düşüncenin temel ilkelerinden biri değer akışıdır ve yalın üretimde işletmede uygulanan faaliyetler ve süreçler değer akışı etrafında yürütülmektedir (Lin ve Quingmin, 2009). Yalın üretim müşteriye odaklanarak, müşterinin istek ve ihtiyaçlarını tam olarak karşılayacak sistem kapasitesini veren üretim dengelenmesini sağlamaktadır (Nomak ve Durmuşoğlu, 2003).

Yalın üretim felsefesinin kaynağı, Toyota Üretim Sistemi veya Just-in-Time olarak bilinen 2. Dünya Savaşı'ndan sonra Toyota Motor Corporation'da üretkenliği artırmak ve maliyetleri düşürmek için doğmuştur. Bununla birlikte, "yalın" terimi ilk olarak Womack, Jones ve Roos tarafından ortaya atılmıştır (Simons ve Zokaei, 2005). İlk israfla mücadeleyi Toyota yöneticisi Taiichi Ohno (1912-1990) yedi *Muda* türünü belirlemekle başlamıştır. Muda Japonca dilinde israf demektir. Bu yedi Muda hatalar, ihtiyaç duyulmayan malların fazla üretilmesi, işlenmeyi ve tüketilmeyi bekleyen parça stokları, gereksiz işlem, gereksiz hareket, gereksiz taşıma ve beklemedir (Womack ve Jones, 2010). Üretim işlemlerinde çeşitli gereksiz nedenleri tek tek incelenmiş ve Toyota'daki deneme yanılma yöntemi ile çözümler

geliştirilmiştir. Toyota'nın yalın üretimdeki amacı, atıkların tamamen ortadan kaldırılarak maliyetlerin düşürülmesi ve talep değişikliklerine esnek bir şekilde adapte olabilmek için fabrikalarda sürekli bir ürün akışı sağlanmasıdır. Gerekli miktarda, gerekli zamanlarda sadece gerekli malzemeleri üretmektir. Değer akış yönetimi (DAY), veri toplayarak ve analiz edilerek oluşturulan değer akış haritaları yardımıyla planlama sürecidir (Simons ve Zokaei, 2005). Bu yöntem, çalışanların istek ve ihtiyaçlarını daha kolay ve kısa sürede karşılamak için yapılan iyileştirmelerin nasıl ve ne zaman gerçekleştirileceğini planlama ve yetkilendirmesini kapsayan sistematik bir yaklaşımdır. Buradaki amaç çalışanların daha hızlı ve daha fazla çalışması değil, müşteri talebine göre malzemenin üretim sistemi boyunca akmasını sağlayan bir sistem kurmaktır (Şahin, 2005). Şekil 27'de tipik bir değer akış yapısı gösterilmiştir.



Şekil 27. Tipik Bir Değer Akışı Yapısı

Kaynak: Baggaley ve Maskell, (2003)

Değer akış yönetiminin amacı, israfı minimuma indirmek, kaliteyi iyileştirmektir, akış süresini ve maliyeti azaltmaktır (Lin ve Quingmin, 2009). İsraf kaynakların etkin ve verimli kullanılamaması, zaman kaybı ve verimsiz süreçlerin varlığından meydana gelmektedir. Süreçlerin israftan arındırılması gerekmektedir.

Yalınlık kavramı hata, stok, fire, maliyet, üretim alanı, kullanıcı memnuniyetsizliği unsurlarının en aza indirgenmesini ifade etmektedir. Yapısında hiç bir gereksiz unsur taşımamaya ve gereksiz unsurlardan tamamen arınmaya odaklı bir ifadedir (Womack ve Jones, 2010).

Yalın düşüncenin amacı, değerin hammadde ile başlayarak verimli bir şekilde akış ile müşteriye teslim edilmesidir. Başarı sağlayabilmek için zamanı verimli kullanmak, gereksiz harcamaları yok etmek ve müşteriler için en iyi değeri oluşturma amacını benimsemek gereklidir (Lin ve Quingmin, 2009).

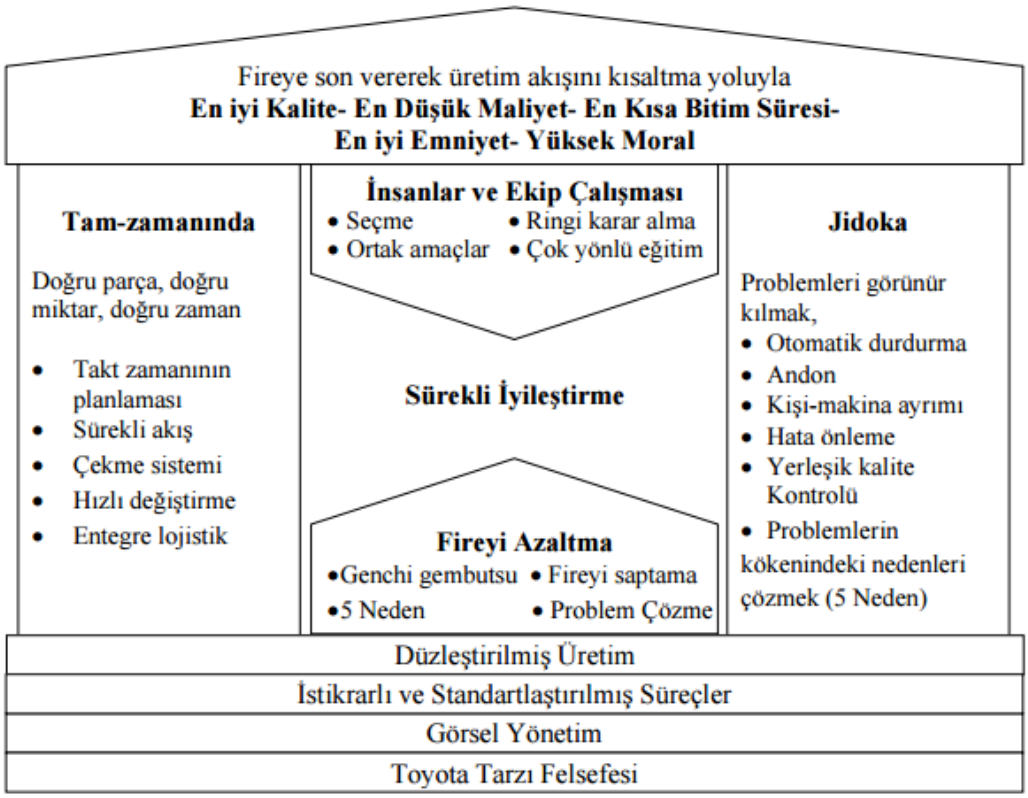
Üretim hızının arttırılıp, akış süresini azaltarak, kalite, maliyet, teslimat performansını aynı anda iyileştirmek yalın üretimin ana stratejisidir (Womack ve Jones, 2010). Yalın bir organizasyonda Tablo 10'de belirtilen adımlardan oluşmaktadır:

Tablo 10. Yalın Uygulama Adımları

Aşamalar	Yapılanlar	Süresi
Başlama	Bir değişim katalizörü bul Yalın bilgisini öğren Değer akışlarını haritalandır Şirket kapsamını genişlet	İlk 6 ay
Yeni bir organizasyon oluştur	Ürün ailesiyle yeniden organize et Bir yalın fonksiyonu oluştur Bir gelişim stratejisi tasarla Mükemmellik felsefesini aşıla	6 aydan 2 yıla kadar
İş sistemleri kur	Yalın muhasebeyi başlat Ödemeyi firma performansıyla ilişkilendir Politika yayılımını başlat Yalın öğrenmeyi başlat Uygun araçları bul	3 yıldan 4 yıla kadar
Uygulamayı tamamla	Önceki adımları tedarikçilere ve müşterilere uygula Global strateji geliştir Yukarıdan aşağıya ve aşağıdan yukarıya ilerlemeleri aktar	5.yılın sonuna kadar

Kaynak: Womack ve Jones, 2010

Yalın üretim sisteminin anlatımı için kullanılan araçlardan biri "Ev" modelidir. Evin çatısını en iyi oranlarda kalite, bitim süresi ve maliyeti oluşturur. Sağda ve solda bulunan iki temel direk "tam zamanında üretim" ve "jidoka" dır. Daha sonra sistemin devamlılığını sağlamak için "heijunka" gerekir. Evin her bir unsuru oldukça önemlidir fakat daha önemlisi bu unsurların birbirlerini nasıl güçlendirdiğidir. Toyota Üretim Sistemi evi Şekil 28'de gösterilmiştir (Liker, 2005).



Şekil 28. Toyota Üretim Sistemi

Kaynak: Liker, (2005)

4.1. Değer Akış Haritası Oluşturma

Değer Akışı Haritalama (DAH) amacı, gelecek durum değer akışının uygulanmasıyla israflara neden olan kaynakları saptamak ve onları ortadan kaldırmaktır (Rother ve Shook, 1999). DAH, akıştaki bilginin, hammaddenin ve nakit akışının görülebilmesini sağlamaktadır (Lin ve Quingmin, 2009) ve akış haritaları oluşturarak değer akışındaki engeller saptanabilmektedir (Maskell ve Baggaley, 2004). DAH, değer akışındaki değeri, israf kaynaklarını ve israfı görmeyi sağlayan bir yöntemdir.

Değer akışı, tek tek prosesleri değil bütünü iyileştirmeyi amaçlar (Dinesh ve Gupta, 2005). Değer Akış Haritalama, sistematik veri yakalama ve analizi yoluyla zayıf girişimleri planlama ve bağlantılandırma sürecidir.

DAH sekiz adımdan oluşur (Tapping, Luyster ve Shuker 2002):

1. Yalınlığı benimsemek,
2. Değer akışını seçmek,
3. Yalınlık hakkındı bilgi edinmek,
4. Mevcut durum haritası oluşturmak,
5. Yalın metrikleri belirlemek,
6. Gelecek durum haritası oluşturmak,
7. Kaizen planları oluşturmak
8. Kaizen planlarını uygulamak.

Hines ve Taylor (2000)'a göre sistemdeki yedi israf:

1. Aşırı üretim,
2. Hatalar,
3. Gereksiz envanter,
4. Uygunsuz işlem
5. Aşırı yükleme,
6. Bekleme,
7. Gereksiz hareket.

Değer akışı haritalandırmada ilk adım müşterilerin algıladığı değerin belirlenmesinden sonra başlatılır, seçilen bir ürün ya da hizmet ailesi için değer akışının tanımlanmasıdır. Kullanıcıdan ve organizasyonların farklı bölümlerinden bilgi taoplanarak mevcut durumun haritası çizilir. Mevcut durum haritası, gelecek durum akışının tasarlanması için gereken bilgiyi sağlamaktadır (Birgün, Gülen ve Özkan, 2006).

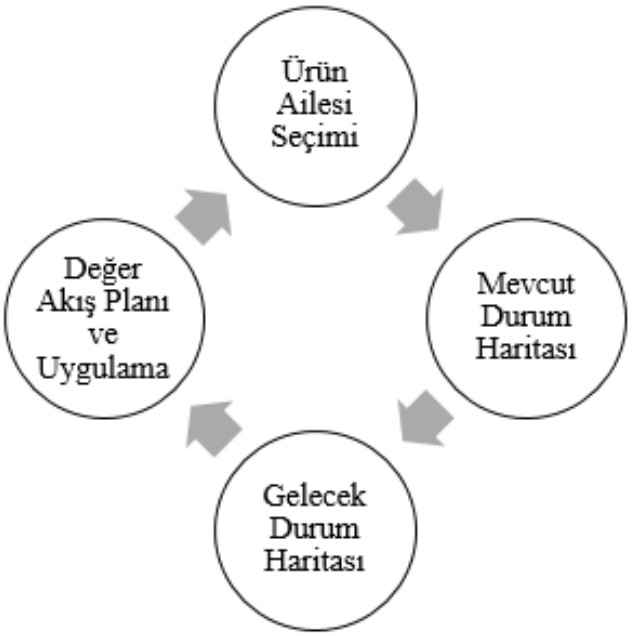
DAH firmalar için gerekli bir araçtır. Çünkü (Yurdugül, 2010):

- Üretimdeki tek proses, kaynak vb. daha fazla görülmesini sağlar. Tüm akışa hakim olunur.
- İsraftan daha fazlası görülür. Haritalama yöntemi ile değer akış yollarındaki israf kaynakları tespit edilir.
- Üretim prosesleriyle ilgili ortak bir konuşma dilinin oluşması sağlanır.

- Üretim akışıyla ilgili kararlar görünür olduğu için tartışmaya açıktır. Aksi takdirde, ortamda alınan birçok kararlar ve detaylar hatalı olabilir.
- DAH bütün akışın nasıl işleyeceğinin tasarlanmasını sağlayarak yalın uygulama için birer plan oluştururlar.
- Akışlar arasındaki ilişkiyi gösterir.

Temin süresi, stok seviyesi, yol alınan mesafe gibi nicel değerler üreten teknikten ve yerleşim planı hazırlamaktan daha faydalı bir tekniktir. DAH, akış oluşturmak için gerekenleri detaylandırarak tanımlanmasını sağlayan nitel bir araçtır. DAH nicel ifadeleri değiştirmek için neler yapılması gerektiğini tanımlamak için vardır. Rother ve Shook (1999)'a göre değer akışı haritalandırma yöntemi Şekil 29'da görüldüğü gibi 4 temel adımdan oluşur. Bunlar:

1. Ürün Ailesinin Seçimi,
2. Mevcut Durum Haritasının Çizimi,
3. Gelecek Durum Haritasının Çizimi,
4. Faaliyet Planının Hazırlanması ve Uygulanmasıdır.



Şekil 29. Değer Akışı Haritalandırma Adımları

Kaynak: Rother ve Shook, (1999)

4.1.1. Ürün Ailesi Seçimi

DAH, ilk olarak oluşturulacak ürün ailesinin seçimi ile başlanmalıdır. Ürün ailesi, birbirine benzer süreç adımlarını takip ederek, üretim esnasında benzer kaynakların kullanıldığı ürün grubudur.

Müşteriler kendi ürünleriyle ilgilendikleri için üretim alanında geçen her adım haritaya eklenmesi gerekmemektedir. Bütün ürün akışlarını tek bir fabrikada göstermek karmaşıklığa yol açabilir (Rother ve Shook, 1999). Ürün aileleri, değer akışının kullanıcısı açısından tanımlanmalıdır. Ürün ailesi, benzer işlem adımlarından geçen ve süreçlerde ortak kaynaklar kullanan ürünler grubudur.

4.1.2. Mevcut Durum Analizi

Mevcut Durum Haritasının amacı, belirlenen süreci ayrıntılı bir şekilde resimlendirerek israfları ortaya çıkarmaktır. Bu resimlendirme işlemi için semboller (EK-A) kullanılmaktadır. Bu haritayı çizebilmek için tedarikçiye, kullanıcıya, çalışma sürelerine, süreç kontrolüne ve süreç adımlarına ilişkin çeşitli bilgilere ihtiyaç duyulur.

Haritadaki en kritik yollardan biri ürünün müşteri tarafından algılanan değerinin açıkça tanımlanmasıdır. Tanımlanmadığı zaman müşteri isteği dışında bir ürün ortaya çıkacağından ekstra iyileştirme masrafı oluşacaktır. Bu yüzden haritalama müşteri isteği ile başlar. Haritalandırmada ilk adım ana hizmet proseslerinin çizimidir. Malzeme akışları yerleşime göre değil, prosesler arası akışa göre çizilir. Bunun için listedeki genel proses bilgilerinden yararlanılır (Rother ve Shook, 1999):

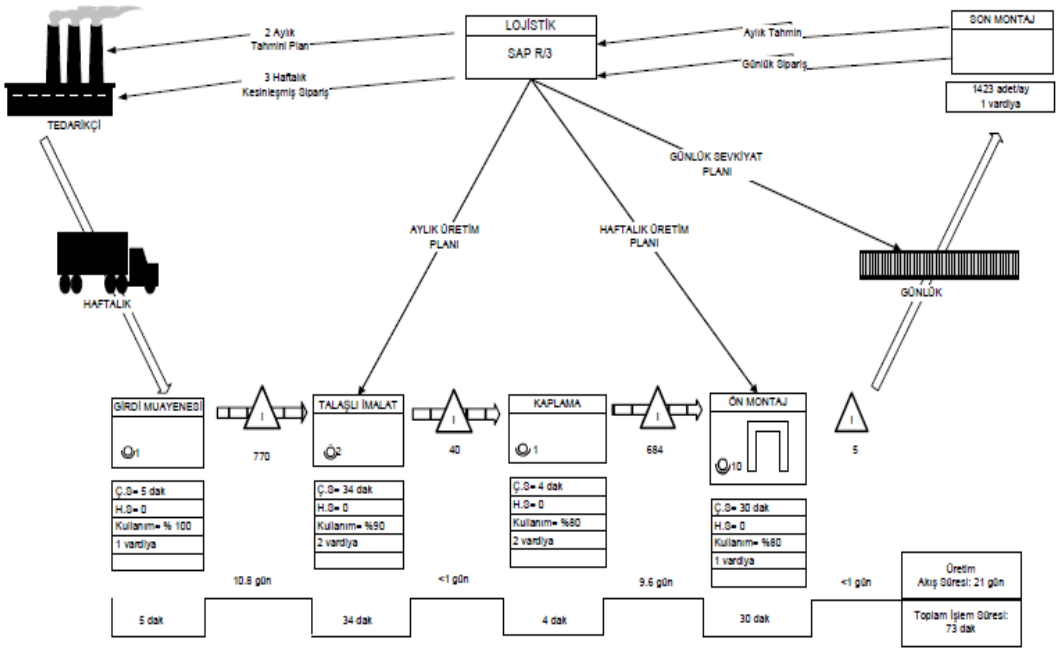
- C/T (Çevrim Süresi - Cycle Time)
- C/O (Model Değiştirme Süresi)
- Uptime (Makine Kullanım Oranı)
- EPE (Every Part Every - Üretim Parti Büyüklüğü)
- Operatör Sayısı
- Ürün/Hizmet Çesitliliği Sayısı

- Çalısma Süreleri (Molalar Hariç)
- Hurda Oranı

DAH üç bölümden oluşmaktadır. Bunlar (Rother ve Shook, 1999):

1. **Süreç ya da Ürün Akışı:** Soldan sağa doğru çizilen tedarikçiler ile başlayıp kullanıcıyla sona eren akıştır.
2. **İletişim ya da Bilgi Akışı:** Formal veya informal iletişimleri, değer katan veya katmayan olmak üzere tüm bilgi akışlarını gösterir.
3. **Zaman Çizgisi ve Ulaşım Mesafeleri (Travel Distances):** DAH altında bulunan, sürece değer katan ve katmayan zamanları gösteren çizgi zaman çizgisidir. Süreç boyunca hareket edilen fiziksel mesafeleri gösteren çizgiye de ulaşım mesafesi çizgisidir.

Örnek olarak hidrolik kapak ürün ailesi için mevcut durum Şekil 30'daki gibi haritalandırılmıştır (Birgün, Gülen ve Özkan, 2006). Haritada, bir parçanın imalatında değer katan faaliyetler 73 dakikadır ve toplam akış süresi 21 gündür. Bu, durum kullanılan zamanın % 99.99'unun israf edildiği göstermektedir. Rother ve Shook (1999), DAH'nın amacının israf kaynaklarının ortaya çıkarılması ve bunların en kısa zamanda gerçekleştirilebilir "Gelecek durum değer akışı" uygulaması ile ortadan kaldırılması olduğunu savunmuşlardır.



Şekil 30. Tamamlanmış Bir Mevcut Durum Haritası Örneği

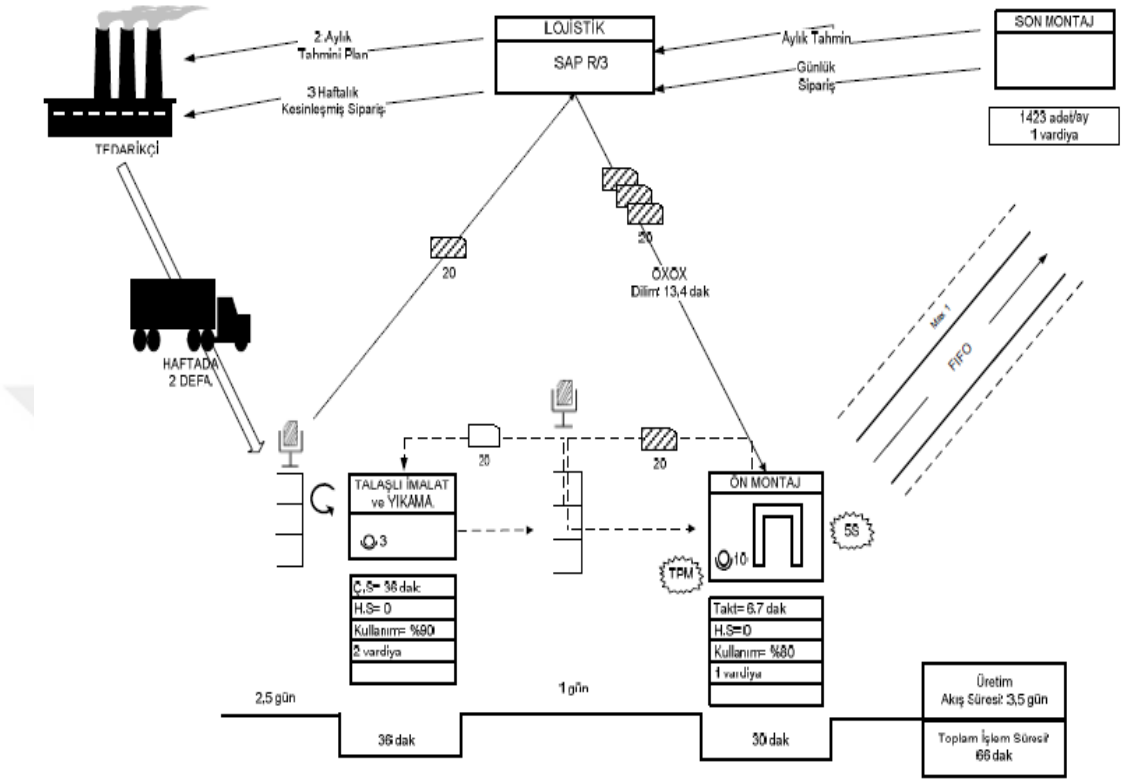
Kaynak: Birgün, Gülen ve Özkan, (2006)

4.1.3. Gelecek Durumun Tasarımı

Gelecek durum haritası, DAH'da bir amaçtır. Hedef yalın bir değer akışı olduğundan şemadaki Gelecek Durum Haritası (GDH) ön planda olmalıdır. GDH oluşturmak için mevcut durum haritası baz alınarak iyileştirme yapılması gereken noktalar tespit edilir ve gelecek durum simgeleri ile işaretlenir. Yapılacak iyileştirmeler belirlenir ve gelecek durum haritası oluşturulur. En son değer akış planı hazırlanır. Belirlenen değişimler gruplandırılarak önceliklendirilir, uygulamaya konulur ve çıkan sonuçlar değerlendirilir. GDH sürekli iyileştirme kapsamında yeni bir mevcut durum haritasının ortaya konmasıdır. Gelecek durumun çizimi sayesinde mevcut durumda gözden kaçan bilgiler ortaya çıkacaktır. Bu plan gelecek duruma nasıl ulaşılacağını kapsar (McDonald, Aken ve Rentes, 2002).

Şekil 31'de görülen Gelecek Durum haritasına göre tedarikçiden döküm halinde gelen hidrolik kapaklar için süpermarket kurulmalıdır ve tedarikçi haftada iki defa sevkiyat yapması sağlanmalıdır. Bu sayede malzeme stokları 10,6 günden 2,5 güne indirilerek %75 oranında bir iyileştirme sağlanacaktır. Ayrıca, tedarikçiyi A

sınıfı kategorisine yükseltmek için girişimde bulunularak girdi muayenesinin ortadan kaldırılmasıyla 5 dakikalık işlem tasarrufu sağlanacak, bekleme ve yığılmalar engellenecektir (Birgün, Gülen ve Özkan, 2006).



Şekil 31. Gelecek Durum Haritası Örneği

Kaynak: Birgün, Gülen ve Özkan, (2006)

4.1.4. Değer Akış Planı ve Uygulama

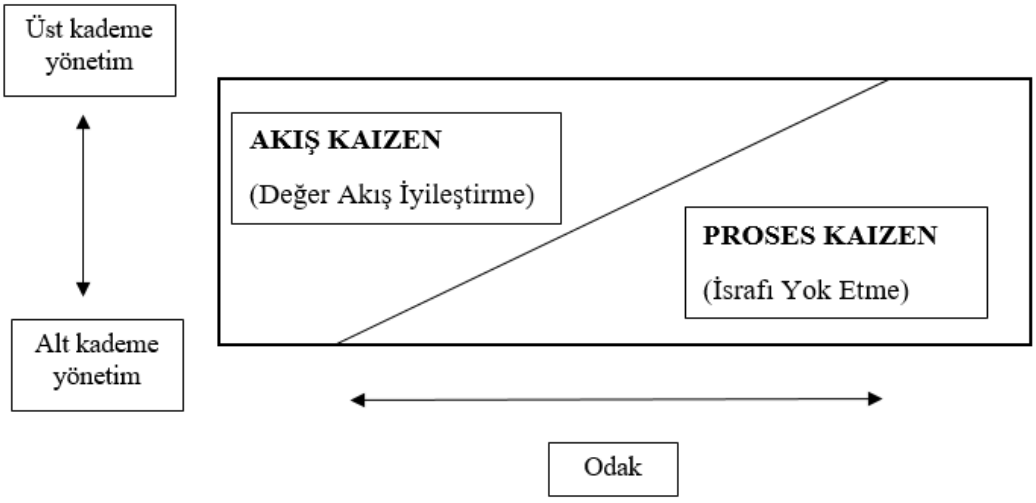
DAH uygulamasının son aşaması tasarlanan gelecek durum haritasının uygulanması için plan oluşturulması ve bu planın uygulamaya geçirilmesidir.

Değer akış planı, gelecek durum haritasında nereye gidilmek istendiğini gösterir.Gerekenler adım adım planlanır, ölçülebilir hedefler konulur (Rother ve Shook, 1999). Planın uygulamasına "hangi sıra ile uygulamalıyız?" ve "nereden başlamalıyız?" sorularıyla başlanmalıdır. Bu belirlenirken çalışanlar tarafından prosesin iyi derecede anlaşıldığı yere, iyileştirmelerin en fazla olacağı yere ve başarı ihtimalinin yüksk olduğu yere öncelik verilmelidir (Rother ve Shook, 1999).

Minimum israflı sürekli akış, aşırı üretimin ortadan kaldırılması demektir. Akışı üretime başlatmak için "çekme"ye ihtiyaç vardır. Birden fazla akış varsa bu akışları yalınlaştırmak için seviyelendirilmeleri gerekmektedir. Seviyelendirmenin olmadığı yerde farklı ürünler partiler halinde üretilmektedir (Rother ve Shook, 1999).

4.2. Değer Akış Haritalama ve Kaizen

DAH için yapılan yalın uygulamada bir çok kişi görev alır. Bu kişilerin değer akış haritalandırmayı anlamaları ve gelecek durum haritalarını okuma yeteneğine sahip olmaları gereklidir. Ancak değer akış haritalama ve gelecek durum uygulama takımlarının, ürünün değer akış sınırlarını görebilen ve orada değişikliklerin gerçekleşmesini sağlayabilen bir kişi tarafından yönetilmelidir. Değer akış iyileştirmeleri yönetim tarafından yapılan Kaizen'dir. Şekil 32'de görüldüğü gibi işletme için hem akış kaizen hem de proses kaizen önemlidir. Değer akış iyileştirme olan akış kaizen'in odağı malzeme ve bilgi akışıdır, takım seviyesinde israfların ortadan kaldırılmasını sağlayan prose kaizen'in ise odağı insan ve proses akışıdır. Kaizen Japonya' nın rekabetteki başarısının anahtarıdır ve her alanda sürekli iyileştirme faaliyetleridir. Japoncada; Kai (değişim) ve zen (daha iyi) kelimelerinin bir araya getirilmesinden oluşan Kaizen kavramı, Türkçe'de anlamı sürekli iyileştirme, sürekli gelişmedir (Zoroğlu, 2013).

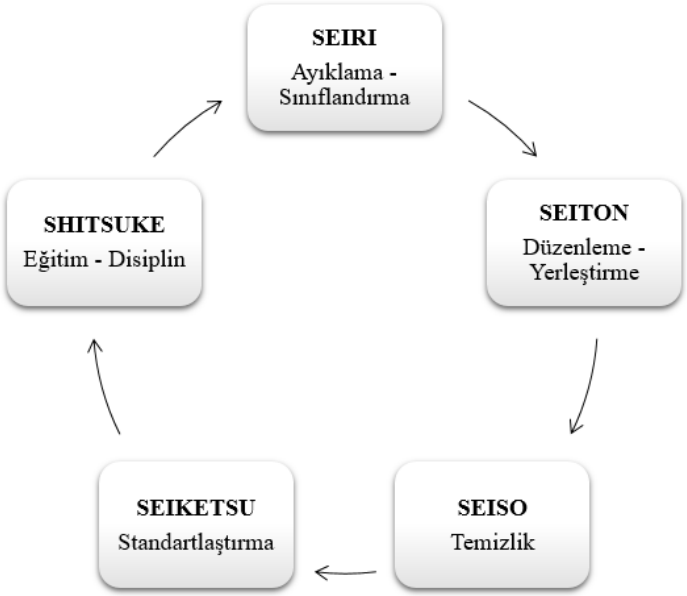


Şekil 32. Kaizen'in iki seviyesi

Kaynak: Zoroğlu, (2013)

4.2.1. 5S Tekniği

Akışı yaratma ve yalınlığı sürdürmede önemli araçlardan biri 5Stir. Yalınlığı uygulamak için gereken temel disiplini yaratmak için ilk olarak 5S'i çalışanların alışkanlık haline getirmesi sağlanmalıdır. Firmalarda israfları azaltmak, sürekliliği sağlamak ve kaliteli bir çalışma ortamı yaratmak için bir takım tekniklerden faydalanılır. Bu tekniklerden biri de 5S'tir. 5S Şekil 33'te de görüldüğü gibi 'S' harfiyle başlayan beş Japonca kelimeden oluşmaktadır (Jackson, 2009).



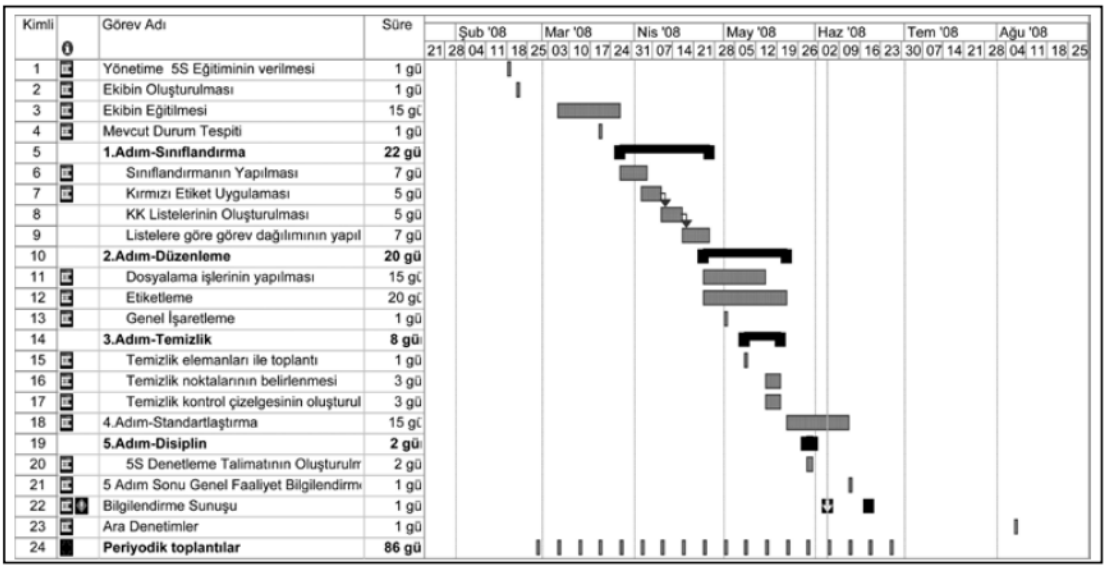
Şekil 33. 5S sistemi

Kaynak: Jackson, (2009)

- **Seiri** (*Ayıklama-Sınıflandırma*) : Çalışma alanında gerekli ve gereksiz nesneleri ayıklamak ve ihtiyaç duyulmayan herşeyi atmak. Seiri aşamasında gerekli ve gereksiz nesneler belirlendikten sonra, gereksiz olan nesneler kırmızı etiketlerle belirlenip, kırmızı alanlara taşınır.
- **Seiton** *(Düzenleme-Yerleştirme)* : Düzen yaratıp bunu sürdürmek. Seiton aşamasında tüm nesnelerin kullanım sıklığına göre düzenlenir ve kolayca bulunacağı şekilde yerleştirilir. Dolaplar ve çekmeceler düzenlenir. Masaüstlerindeki gereksiz eşyalar ayıklanır, gerekli olanlar ise düzenlenir. Dosyalar sıralı ve numaralandırılmış bir şekilde dolaplara dizilir. Etrafta risk taşıyan ve çirkin bir görüntü oluşturan kablolar iş güvenliğine uygun olarak yeniden düzenlenir.

- **Seiso** *(Temizlik)* : Temizlemek. Seiso aşamasında tüm bu yapılan düzenleme ve ayıklama aşamalarında ofisin temizliği de sağlanmaktadır.
- **Seiketsu** *(Standartlaştırma)* : İlk üç S'yi sürdürmek için kurallar oluşturmak,
- **Shitsuke** *(Eğitim-Disiplin)* : İlk dört S'nin disiplinini devam ettirmek (Byrne, 2015).

5S uygulanacak çalışma ortamı için ekip üyelerine eğitim sürecinden geçerler. Eğitimde projenin amacı, uygulama basamakları, sorumlulukları ve çıktıları aktarılır. Projenin aşamaları ve süreleri belirlenerek aşağıdaki verilen Şekil 34'teki gibi bir Gantt şeması oluşturulur:



Şekil 34. Örnek 5S Uygulama Projesi İş-Zaman Şeması

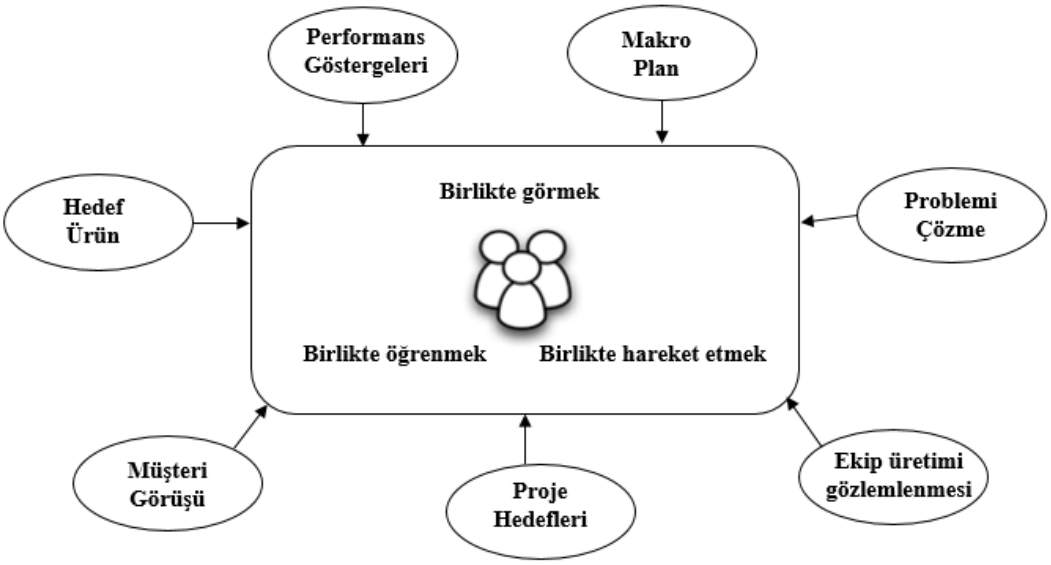
Kaynak: Doğan, Özkütük ve Doğan, (2014)

4.2.2. OBEYA Tekniği

OBEYA kelime olarak Japoncada "Geniş Oda" anlamına denk gelmektedir. Bir projenin görsel sunumlarıyla (çizelgeler, diyagramlar, çizimler ve metrikler gibi) kaplandığı bir odaya atıfta bulunulur. Etkin iletişimi, zaman tasarrufunu arttırmak için görsel şekiller ve grafiklerle birlikte kullanılan proje yönetim aracıdır. Bir BT

projesinde, üretim akışını görselleştirmek için bir sistem olmalıdır. Proje yöneticisi ve proje ekibi, OBEYA odasını aşağıdaki öğelerle inşa etmeye başlarlar:

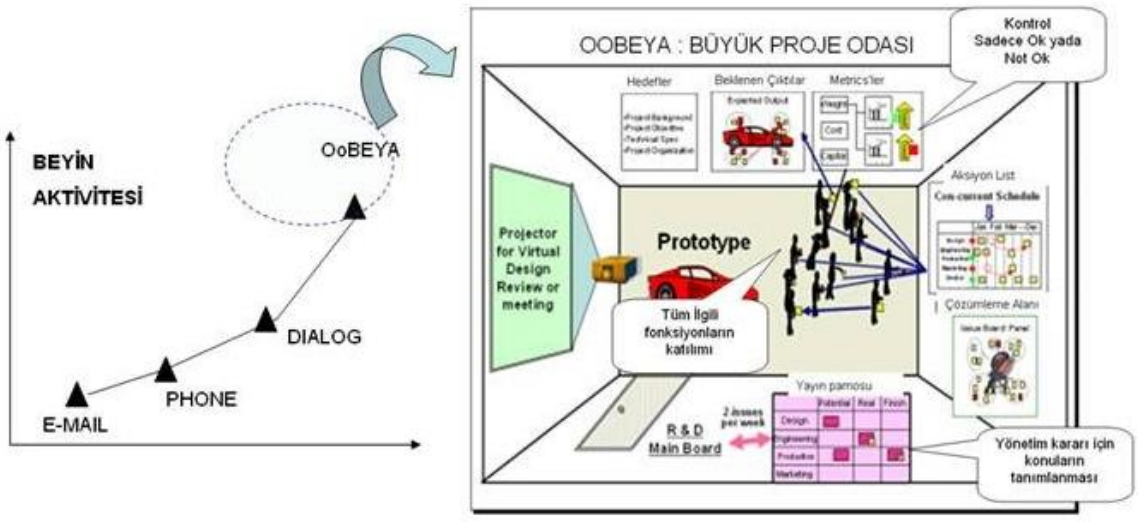
- Proje Hedefleri proje geçmişini ve vizyonunu tanımlamaktadır.
- Müşteri Görüşü, ürün tasarımının temelini oluşturan müşteri tercihlerini tanımlar.
- Hedef ürün nihai çıktıyı ve ayrıca ürünün mevcut durumunu gösterir.
- Performans Göstergeleri, proje hedeflerini ve müşterinin görüşünü 3 ile 4 ana göstergede (müşteri memnuniyeti, kalite, zaman, maliyet) belirtir.
- Makro Plan, tüm projenin önemli çıktılarını ve kilometre taşlarını bir araya getirir.
- Problem Çözme, devam etmekte olan tüm sorunları ve belgeleri nasıl çözüldüklerini listeler.
- Ekip üretimi gözlemlenmesi, her takım için üretim akışını gösterir.



Şekil 35. OBEYA Yönetim Aracı Oluşumu

Kaynak: www.lutfiapiliogullari.com, 10.02.2017

Şekil 36'da görüldüğü gibi OBEYA, büyük toplantı odasında tek bir sonuca odaklanmış, hiyerarşinin olmadığı, sorumluluğun ekip üyelerinde olduğu çözüm odaklı proje toplantılarıdır.



Şekil 36. OBEYA Konsepti (Ürün geliştirme ve yeni ürün devreye alma süreci)

Kaynak: www.lutfiapiliogullari.com, 10.02.2017

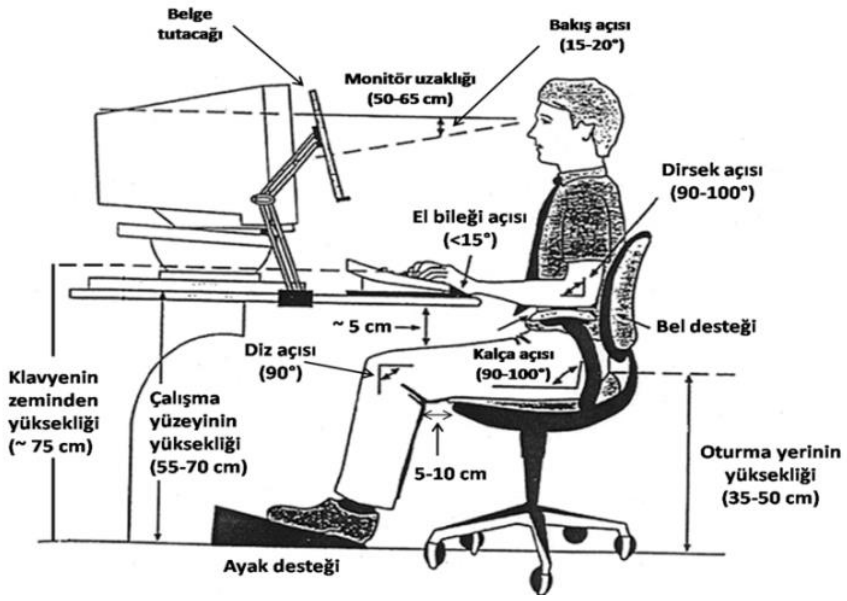
4.2.3. Metot Etüdü Tekniği

Metot etüdü, firmalarda ya da iş yapılan bir ortamda, işlerin daha iyi nasıl yapılabileceği ile ilgilenir. Yapılma amacı her türlü israfın önlenmesini sağlamaktır. Metot etüdünün hedefi, gereksiz kaynakları ortadan kaldırmaktır (Doğruer, 2014). Metot etüdü konuları;

- Süreçlerin,yöntemlerin ve faaliyetlerin düzeltilmesi, düzenlenmesi,
- Fabrikada, atölyede ya da ofiste işin yapıldığı yerin düzeninin, tesisat ve donanım tasarımlarının düzeltilmesi: tasarımının ergonomik olması,
- İnsan gücünde artış sağlanması ve aşırı yorgunluğun azaltılması,
- İşgücü, makine, malzeme kullanımının düzeltilmesi, düzenlenmesi,
- Daha iyi çalışma koşullarının geliştirilmesi.

Süreç şemalarında herhangi bir işin ya da işlemin kaydedilmesi için beş standart simge (EK-B) kullanılır. Yerleşim düzeni metot etüdünde oldukça önemlidir. Yerleşim düzenlemesi yapılmayan ofis ya da atölyelerde iş akışında malzeme izlemede zorluklar, gereksiz taşıma, işlem ve bekleme sürelerinde artış, alandan tam olarak yararlanamama, çalışanın bedensel ve zihinsel yorgunluğuna bağlı dikkat

eksikliğinin artmasına neden olur. Kullanılan araç ve gereçler düzen içinde yerleştirilmelidir. Çalışanın hareketlerini kolay kontrol edebilmesi için çalıştığı alanın ve kullandığı araçların ergonomik olması önemlidir. Çalışılan ortamın esnek yerleşim düzeni olmalıdır. Üretim faaliyetleri de dengeli biçimde dağıtılmalıdır. Örneğin, Yazılım projelerinde de bu yöntem kullanılarak proje teslim süresi kısaltılabilir. Örneğin yazılım geliştiriciler işleri gereği saatlerce ekran başında kod yazmaktadır ve bu çeşitli sağlık sorunlarına neden olmaktadır. Oturduğu masa düzeni Şekil 37'deki gibi egonomik olursa sağlık sorunları minimuma indirilmiş olur.



Şekil 37. Ergonomik Ofis Masa Düzeni

Kaynak: www.ofisteyasam.com, 11.02.2017

5. LİTERATÜR TARAMASI

Süreç iyileştirme, yalın üretim, değer akış haritalama, yazılım süreçleri ve yazılım süreç iyileştirme konularına ait çok sayıda çalışma bulunmaktadır. Yapılmış bazı araştırmalar ve çalışmalar, aşağıda kısaca özetlenmiştir.

Sönmez (2013) çalışmasında, 6 Sigma kullanarak kalkınma ajansında bir iyileştirme projesi uygulamıştır. Uygulamada yapılan yatırım teşvik belgelerinin düzenlenmesinde gerçekleşen süreçlerin iyileştirilmesini hedeflemiştir. Pozitif ve negatif yönleriyle yapılan bu süreç iyileştirme çalışmasında sürecin %95 güven aralığında iyileştirildiği ANOVA testi uygulanarak gösterilmiştir.

Birgün, Gülen ve Özkan (2006), traktör üretimi yapan bir fabrikada müşteri beklentilerini karşılamak üzere yalın üretim prosesi dahilinde değer akış haritalama çalışmasında bulunmuşlardır. Oluşturulan değer akış haritasında kullanılan zamanın % 99.99'unun israf edildiği saptanmıştır. Çalışmanın sonunda alınan sonuçlar firma yetkilileri ile paylaşılmış ve bu sonuçlara göre çekme sisteminin kurulmasıyla temin süresinin 21 günden 3,5 güne kısalması ve envanter devrinin 6 kat artması mümkün olacağı öngörülmüştür. Ayrıca simülasyon sistemi ile tedarik sürecinin haftada iki kez olmasının sistemde yaratacağı etkinin görülmesine fayda sağlayacağı önerisi getirilmiştir.

Özveri ve Güçlü (2015) çalışmasında, işletmelerde yaygın olarak kullanılan süreç yönetimi tekniği ile işlerin akışı etkin olarak yönetilebileceğini, süreçlerde kullanılan kaynakların değerlendirilmesi ve verimliliklerinin sorgulanması gerektiğini savunmuşlardır. Uygulamalarında değer akış haritalarında iyileştirme noktalarının seçiminde Analitik Hiyerarşi Süreci (AHP)'nin nasıl kullanılabileceği değerlendirilmiştir. Ayakkabı sektöründe faaliyet gösteren bir işletme için mevcut durum değer akış haritası çizilmiştir. Çizilen mevcut durum haritası analiz edilerek uygulamanın yapıldığı firmada iyileştirilmesi düşünülen noktalar belirlenmiş ve AHP yöntemi uygulanarak, iyileştirilecek noktalar seçilmiştir. Gerekli iyileştirmeler

yapıldıktan sonra sonuçlar analiz edilmiştir. Analiz raporuna göre firmada uygulanan iyileştirmelerin olumlu sonuçlar verdiği tespit edilmiştir.

Seth ve Gupta (2005), otomotiv sektöründe faaliyet gösteren bir firmada süreç iyileştirmesi için değer akış haritalamayı kullanmışlardır. Çalışmalarının sonucunda kişi başına üretimde artışla verimliliği etkileyen üretim içi stoklarda ve nihai ürün stoğunda azalma olduğunu öne sürmüşlerdir.

Çırak (2013) çalışmasında, proje yönetiminde yalın ve kısıtlar teorisini incelemiştir. Bir ambalaj makineleri üreticisinin proje yönetimi ve planlama süreçleriyle ilgili uygulama yapmıştır. Değer Akışı Haritalama yöntemi kullanmıştır. Çalışmanın devamında Yalın Üretim tekniklerinin kullanılmasıyla elde edilecek kazanımlar tespit edilmiş ve yeniden planlanmıştır.

Chitturi, Glew ve Paulls (2007), mevcut durum değer akışı haritasının geliştirilmesi aşamasını araştırmışlar ve gelecekte ulaşılmak istenen değer akışın nasıl gerçekleştirilmesi geretiğiyle ilgili konulara değinmişlerdir. Seth ve Goel (2007), Hindistan'da pamuk yağı üretimi yapan bir fabrikada bekleme sürelerinin ve israfların ortadan kaldırılması için değer akışı haritalama yöntemini kullanmışlardır. Prabhu, Surekha, Holla ve Patel (2008), Hintli bir nakliye firmasının iş süreçlerinin iyileştirilmesi ve değer katmayan faaliyetleri belirleyerek süreçlerin iyileştirilmesi amacıyla değer akışı haritalama yöntemi kullanmışlardır.

Tunç (2016) çalışmasında, yalın yönetimi incelemiştir. Sosyal Güvenlik Kurumu'na yalın yönetim uygulamıştır. Kurumlarda yalın yönetimle ilgili vizyon ve kültürün oluşması için önerilerde bulunmuştur. Planlama, iletişim ve problem çözme için ortak bir lisan oluşturulmasını savunmuştur.

Yaman (2007) çalışmasında, yalın yönetim temel ilke ve prensiplerini incelemiştir. Eskişehir Askeri Havacılık kurumunda 212 kişiden topladığı verilerle hiyerarşiye dair analizler yapmıştır. Kar amacı olmayan kurumlar için yalın yönetimi uygulamaya geçirmek kar amaçlı sistemlere göre daha zor olduğunu savunmuştur ve kurum yalın yönetimin en kolay nasıl uygulanabilirliğiyle ilgili önerilerde bulunmuştur.

Azizia (2015) araştırmalarında, değer akış haritalama yöntemi kullanarak küçük ve orta büyüklükteki işletmelerde verimliliği arttırmak için israfların ortadan kaldırılması ve performans artışını hedeflemiştir. Çalışmalarında mevcut durum haritasına göre bekleme sürelerini ve israfları tespit etmiştir. Kaizen iyileştirme aktivitelerini kullanarak hazırlanan gelecek durum haritası belirlemiştir. Böylece üretim süresini azaltarak işletmede iyileştirme sağlanmıştır.

Yurdugül (2010) çalışmasında, yalın üretim ve değer akış haritalama yöntemi üzerinde durmuştur. Eskişehir'de bulunan Camiş Ambalaj fabrikasında kutu üretim grubuna, değer akış haritalandırma uygulaması yapılmış, gelecek durum haritası oluşturmuştur. Uygulamada üretim düzgünleştirme yöntemi kullanarak üretim planı oluşturmuştur ve gelecek durum haritası için çekme sistemini kullanarak kanban sistemlerini de içeren bir tasarım yapmıştır. Uygulamanın sonunda genel bir analiz ve değerlendirmesini yapmıştır. Stok seviyesini %75 oranında azaltıp 39 günden 10 güne indirmiştir. Makinalar arasındaki 8,51 günlük bekleme süresini ortadan kaldırmıştır. Kutu için işlem süresinde %72 oranında iyileştirme sağlamıştır.

Yazılıma özgü ilk süreç modeli, ABD Savunma Bakanlığı desteğiyle kurulan Carnegie Mellon Üniversitesinde Yazılım Mühendisliği Enstitüsü'nde geliştirilen Yetenek Olgunluk Modeli, CMM olmuştur. 1995 yılında ilk versiyonu ISO 15504 teknik rapor olarak yayınlanmıştır. 1998 yılında üçüncü versiyonu, resmi bir ISO yazılım süreç değerlendirme standardı olarak (ISO/IEC, 2006) yayınlanmıştır.

Nalbant (2012) makalesinde, yazılım firmalarında kullanımı için yazılım geliştirme sürecinin otomasyonunu sağlayan bir bilgi sistemi önermiştir. Bu otomasyon sistemi yazılım projelerinin planlanmasını, yönetilmesini, kontrolünü, ölçülmesini ve iyileştirilmesini sağlamıştır. Bu sistem sayesinde yazılım geliştirme süreçlerini daha verimli hale getirebileceklerini savunmuştur ve fayda sağlamayı amaçlayan bir model önermiştir.

Abe, Mizuno, Kikuno, Kikuchi ve Hirayama (2006) çalışmalarında, yazılım metodolojilerinde, süreç iyileştirme çalışmalarını nesnel bir temele dayandırmak ve kontrolünü sağlamak için istatistiksel metotlar önermişlerdir. Yazılım proje

çıktılarının tahmini üzerindeki araştırmalarında, daha fazla başarıyı yakalamak için yazılım karakteristiklerinin belirlenmesine odaklanmışlardır.

Çetin ve Durdu (2015)'nun Çevik yazılım geliştirme yöntemlerinin kullanımını ve vaat ettiği avantajları içeren güncel durumunu belirlemeye yönelik çeşitli çalışmaları bulunmaktadır. Bu çalışmalarında, ülkemizde yazılım sektöründeki firmalarda çevik ve geleneksel yaklaşımların ne kadar kullanıldığına dair bir anket hazırlayarak 74 firmadan veri toplamışlardır. Uygulanan anket sonuçlarına göre oranların birbirine yakın olduğu ve giderek çevik yaklaşımlara yönelimin arttığı gözlemlenmiştir. Yazılımcılar çevik yaklaşımların verimliliği, kaliteyi ve müşteri memnuniyetini arttırdığını belirtmişlerdir.

Başar, Özkaya ve Kesgin (2014), bir teknoloji şirketinde Şelale yönteminden, Çevik yönteme geçiş uygulaması yapmışlardır. Çevik yöntemlerden SCRUM metodolojisi benimsenmiştir. Firmanın tüm yazılım geliştirme aşamalarında 5 yıl boyunca SCRUM uygulanmıştır. Fakat zaman içerisinde bu yöntemin kullanımında çeşitli sorunlarla karşılaşılmış, bu sebeple KANBAN'a geçilmesi planlanmıştır. KANBAN metodunun firmaya uyarlanmasında, SCRUM yönteminin kullanımından elde edilen deneyimler kullanılmıştır. Kanban yöntemiyle alınan ilk sonuçlar kalite artışını gösterdiği saptanmıştır ve artık bu metodun kullanılmasının firma açısından daha faydalı olacağı önerilmiştir.

Güven (2015) çalışmasında, bilişim sektöründe yalın metodolojinin önemi üzerinde durmuştur. Projelerdeki başarısızlıkların kaynağı yönetimden değil sistemden kaynaklı olduğunu savunmuştur. Firmada, hızlı bir şekilde ve en az kaynak israfı ile geliştirilecek olan yazılımın başarıya yalın metodolojilerle ulaşılacağını ispatlamıştır.

Gül (2006) yaptığı çalışmada, yazılım geliştirme süreci temel yönleriyle incelenmiştir. Yazılım geliştirme metotları, yazılım geliştirme sürecindeki temel aktiviteler, proje yönetim süreçleri, yazılım geliştirme sürecinin iyileştirilmesi ve modelleri, proje başarısızlık sebepleri, başarı faktörlerini uygulama üzerinde anlatmıştır. Türkiye'de yazılım üreten firmalara yazılım geliştirme süreçlerine dair anket oluşturmuş ve bu anketten elde ettiği bulguları değerlendirilmiştir. Anket

değerlendirmesine göre yazılım geliştirme sürecinde eksik ve iyileştirilmesi gereken noktaları saptamıştır. Yönetim desteğinin ve iyileştirme çalışmalarına bir proje gözüyle bakılması kriterlerinin önemli olduğunu belirtmiş, çeşitli önerilerde bulunmuştur.

Meriç (2011) çalışmasında, Yalın Üretim ile Kurumsal Kaynak Planlaması yazılımlarının bütünleştirilmesini incelemiştir. Akademisyen ve uzmanlara yaptığı anket çalışması ile Yalın Kurumsal Kaynak Planlama konusunda görüşlerini araştırmıştır. Tasarım önerileri ve detay tasarım çalışmaları yapmıştır. Yalın Üretim ve Kurumsal Kaynak Planlama sistemlerinin ortak yönlerini ve firmalara sağladıkları avantajları belirlemiştir.

Çandur (2010) bankacılık sektöründe BT alt yapı sisteminin kurulmasına dair çalışma yapmıştır. Proje ve proje ekibinin oluşturulması, BT alt yapı sisteminin planlanması, proje başarısızlıkları ve çözüm önerileri, bölüm ve görev tanımlamaları, bankacılık BT proje planlama ve yönetimi konularını ele almıştır. MS Project programını kullanarak sistemin planlanmasını sağlamıştır. Proje yönetiminde iyileştirme çalışmaları yapmıştır ve MS Project programının bankacılık proje yönetimi konularında uygulama yöntemlerine katkı sağladığını savunmuştur.

6. BİLİŞİM TEKNOLOJİLERİ SEKTÖRÜNDE BİR UYGULAMA

1990'lı yıllardan itibaren Bilişim teknolojilerinin hızla gelişmesi bütün sektörleri bilişim teknolojilerini etkin bir şekilde kullanmaya mecbur kılmıştır. Bu doğrultuda firmalar BT ihtiyaçlarını karşılamak farklı alanlara yatırım yapmaya başlamışlardır.

Günümüzde tüm alanlarda büyük değişim yaşanmaktadır. Özellikle 1980'li yıllarda ekonomide yaşanan önemli değişim ve dalgalanmalar; firmaları yeni arayışlara yönlendirmiştir. 1980'lere kadar bilgi, önemli bir varlık olarak düşünülmemekteydi fakat gelişen bilgi teknolojileri, dünyayı bir ağ sistemiyle donatıp, zaman ve sınır engellerini kaldırarak bilginin vazgeçilmez bir unsur olmasına yol açmıştır. Bilişim sektörüne yapılan yatırım tutarının giderek artması, firmaların bilişim sistemlerine yönelik faktörler üzerinde önemle durmasına sebep olmuştur (Rhinesmith, 2000).

Burada bir BT firması olan Ekho-Tec A.Ş.'nin yazılım sürecinin geliştirilmesi amacıyla bir uygulama yapılmıştır. Firma; Yazılım Geliştirme, İş ve Sistem Analizi, Test Mühendisliği, Proje Yönetimi, Operasyon Destek, Kurumsal Mimari, Sürüm ve Konfigürasyon Yönetimi ve Sistem Entegrasyon alanlarında hizmet vermektedir. Müşteri odaklı hizmet verme yaklaşımı ile kapsamlı yazılım çözümleri sunan Ekho-Tec, müşterileri ile ortak proje ekibi oluşturarak anahtar teslim projeler geliştirmektedir. İş Akışı ve Doküman Yönetimi gibi hazır ürünlerin yanında anahtar teslim projelerle müşterilerin ihtiyaç duyduğu yazılım çözümlerini üreten firma, sigorta sektörü yazılımlarından, telekomünikasyon ve e-ticaret yazılımlarına kadar geniş bir alanda büyük ölçekli yazılım projeleri gerçekleştirmektedir.

Çalışmanın yapıldığı firma sigorta, telekomünikasyon ve e-ticaret alanlarında 12 firmaya hizmet vermektedir. 150 beyaz yakalı personel istihdam edilmektedir. Firmaların istekleri doğrultusunda yıllık veya aylık sözleşmeli ve istenilen alanlarda en az 2 yıl tecrübeli personel istihdamı sağlanmaktadır.

6.1. Değer Akış Haritalama ve Süreç İyileştirme Adımları

Bilişim projelerinin pahalı, uzun zaman gerektiren ve çok emek isteyen projeler olduğundan bir takım sorunlar yaşanmaktadır. Bunlardan bazıları:

- İstihdam edilen çalışanın alanında yetersiz olması,
- Proje için uygun ekip kurulamaması,
- Firmaların ürün tesliminin son teslim tarihinden önce olması için baskı uygulamaları,
- Müşteri firmaların sürekli isteklerini değiştirmesi,
- Müşteri isteklerinin doğru analiz edilememesi ve müşteri ile iletişimden kaçılması,
- Gerekli bütçe ve kaynakların oluşturulamaması,
- Proje yönetim metotlarının uygulanmaması,
- Yanlış teknoloji ve mimari seçimidir.

Şirketin yönetimsel sorunları projelerin süreçlerini sıkıntıya sokmaktadır ve geç teslim edilen ya da teslim edilemeyen işlere neden olmaktadır. Bu çalışmanın amacı bu sorunlara yalın felsefe ve çevik metotla, değer akış haritalama yöntemi uygulanarak çözüm bulunmasıdır.

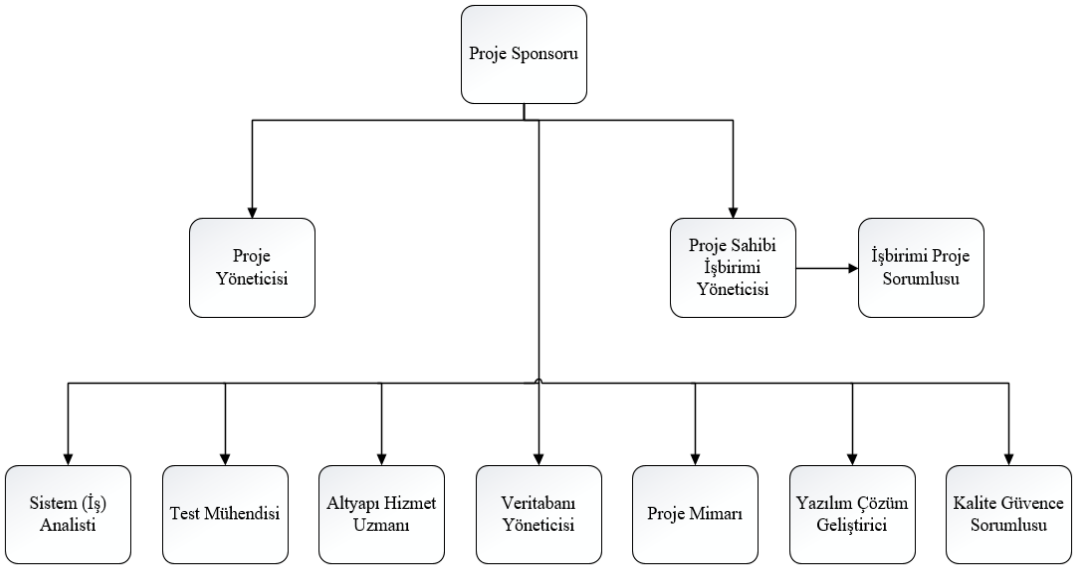
6.1.1. Yazılım Proje Ekibin Kurulması

Uygulama için kullanıcılardan aylık periyodlarda istek ve ihtiyaçlarına göre belirlenen büyüklüklerine göre talepler alınır. Gelen taleplerin süresinin uzun olması halinde, talep alt kısımlara ayrılarak aylık sürümler şeklinde kullanıcıya sunulur. Talebin amacının dışına çıkmasının önündeki en büyük engel, çalışan bir ürün olmasıdır. Aksi takdirde bitmeyen analiz ve planlama süreçleri kısır döngüye dönüşür. Kişilerin yapacağı tüm faaliyetler proje planında yer alır. Proje paydaşları aşağıdaki gibidir:

- Proje Sponsoru
- Proje Sahibi İş Birimi Yöneticisi
- Proje Yöneticisi
- Kalite Güvence Sorumlusu

- İş Birimi Proje Sorumlusu
- Sistem (İş) Analisti
- Test Mühendisi
- Altyapı Hizmet Uzmanı
- Veritabanı Yöneticisi
- Proje Mimarı
- Yazılım Çözüm Geliştirme Sorumlusu

Proje paydaşlarının organizasyon yeri ve kime bağlı oldukları Şekil 38'de gösterilmiştir.



Şekil 38. Proje İçin Oluşturulan Organizasyon Şeması

6.1.2. Yazılım Proje Sürecinin İşleyişi ve Ürün Ailesi Seçimi

Yazılım geliştirme ile alakalı sistem altyapısı için yapılması zorunlu olan program kurulum, belgeleme ve kullanıcı eğitimleri gibi faaliyetler planda belirtilir. Tatil ve izin süreleri de plana eklenir. Aksi takdirde proje süresinde aksamalar yol açar. Kullanıcı talepleri uygulama ürün ailesi olarak seçilmiştir. Her bir talep yazılım projesi adı altında değerlendirilmiştir. Yazılım projeleri; Bütçeleme, Bütçe Onayı Alma ve Sponsor Atama, Gereksinim Analizi Hazırlama, Tasarlama, Kodlama, Test

Etme, Kullanıcı Test Etme, Kullanıcı Onaylama ve Devreye Alma aşamalarından oluşmaktadır. Tablo 11'de bu görevler ve detayları açıklanmıştır.

Bütçeleme aşamasında talep için bütçe yapılacak faaliyetler ele alınır. Tüm faaliyetler bütçede gösterilmelir. Önceki taleplerle benzerlik kontrolü ve piyasa araştırması yapılarak, taslak bütçe oluşturulur. Proje kapsamında tedarik edilecek kaynaklar ve bunlara ilişkin maliyetler belirlenerek nihai bütçe raporu oluşturulur.

Bütçe Onayı Alma ve Sponsor Atama aşamasında nihai bütçe raporu üst yönetime sunulur. Onay aşamasını geçtikten sonra proje için sponsor ataması yapılır. Atanan sponsor sayesinde projenin süreçleri üst yönetim tarafından takibe alınır. Onay sürecinden geçen ve sponsor atanan projenin analiz aşamasında kullanıcının istek ve ihtiyaçları detaylandırılır.

Gereksinim Analizi Hazırlama aşamasında, onay sürecinden geçen ve sponsor ataması tamamlanan projeler iş analistlerine paylaştırılır. Her iş analisti kendisine atanan projeyle ilgili gereksinim analiz dokümanı oluşturulur. İşin büyüklüğüne göre 1-5 gün içerisinde tamamlanması gereklidir. Eğitim gerektiren bir ürün çıkışı olacaksa kullanıcılar için doküman hazırlanır.

Yazılım Geliştirme Süreci, **Tasarlama ve Kodlama** aşamaları gibi temel aşamalardan oluşur. Ekran tasarımındaki ayrıntı düzeyi ile veri model ve rapor tasarımlarındaki düzey farklıdır ancak planlama şekilleri benzerdir. Bu projede yazılım geliştirme ekibi 5 kişiden oluşmaktadır. Teknik mimarlar ekran tasarımını hazırlamışlardır. Yazılımcılar ise ürün için Java platformunda kod geliştirmişlerdir. Tasarlama ve Kodlama görevleri alt sistemlere ayrılır. Alt sistemler, tasarımı basitleştirmek, testleri planlamak ve tekrar kullanım için fayda sağlamaktadır. Yapılan geliştirmelere yazılımcılar tarafından yük, güvenlik ve birim testleri yapılır. Sorun çıkmadığı takdirde kodlar test ortamına aktarılır. Tasarlama ve kodlama aşamaları talebin büyüklüğüne göre 5-18 gün aralığında tamamlanmalı ve test ortamında teste gönderilmelidir.

Test aşamasında sorumlu test mühendisi taleple ilgili test senaryoları ve test planını hazırlar. Bu projede test senaryoları hazırlandıktan sonra adım koşumu için HP ALM (Hewlett Packard, versiyon: 12.50) programı kullanılır. Senaryoların hepsi HP ALM programına yüklenir. Bulunan hatalar program üzerinden derecelendirilerek yazılımcıya gönderilir. Hata çözüm test – yazılım sirkülasyonu bu program üzerinden takip edilir ve günlük adım koşumu yöneticilere düzenli olarak raporlanır. Projede veri, denetim, giriş-çıkış, arayüz ve belleğe bağlı hatalar oluşabilir.

Test mühendisi hataların çözümünün ardından test ortamında test koşumunu tamamlar. Regresyon test adımları belirlenir. Canlı ortam çıkışında sistem üzerinde yapılan her değişiklikten sonra belirlenen regresyon adımları ile sistemin bütünlüğü test edilir. Hatalar çözüldükten sonra aynı adımlar Retest adı altında tekrar koşulur ve tüm adımlar sorunsuz koşulduktan sonra yazılım ekibi test ortamındaki kodları UAT ortamına taşır. UAT ortamı son kullanıcı testlerinin yapılacağı test ortamıdır. Talebin büyüklüğüne göre test aşaması 1-3 günde tamamlanır.

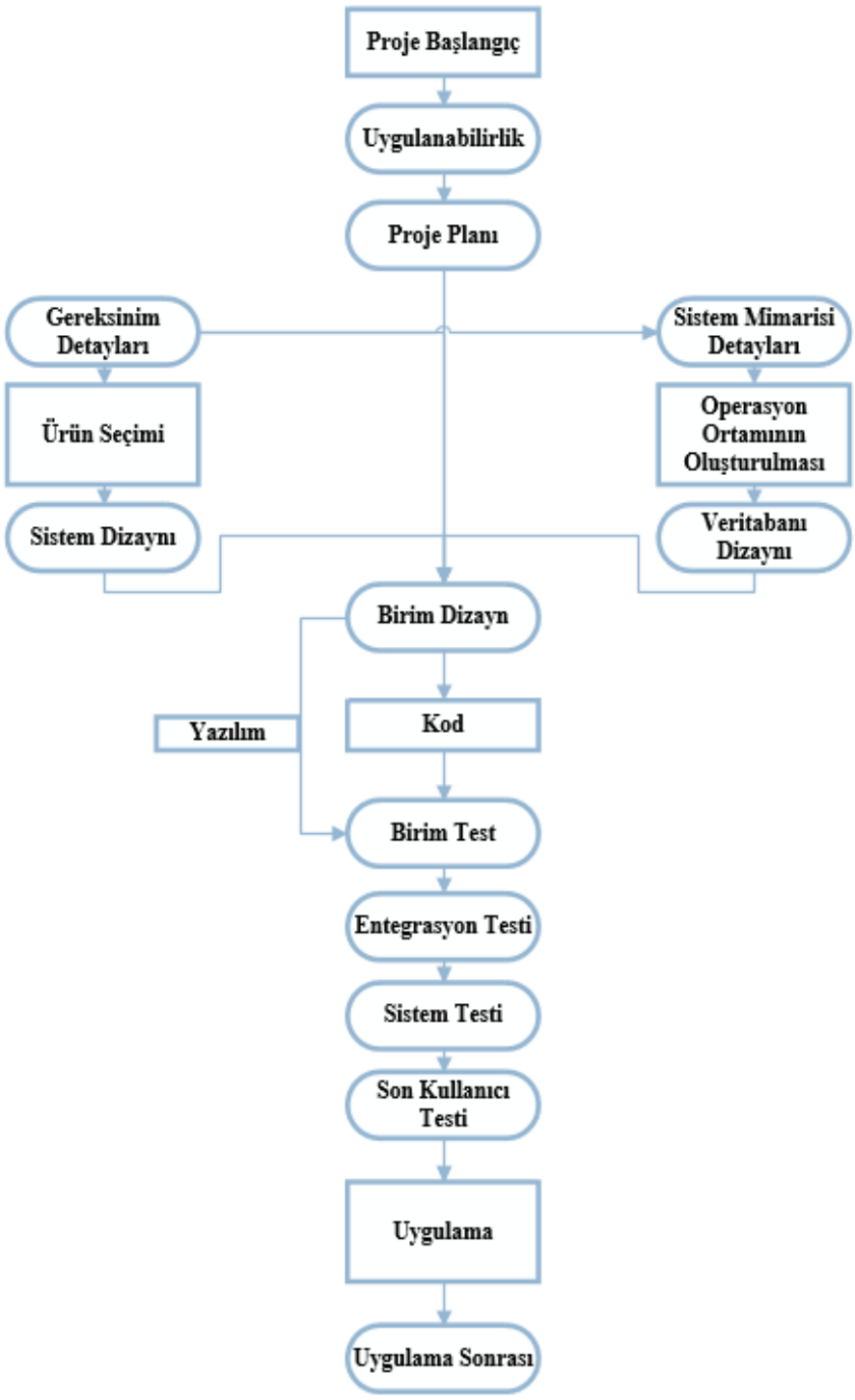
Kullanıcı Test Etme aşamasında, kullanıcı UAT ortamında kendi oluşturduğu senaryolarla ürünü test eder. İstek ve ihtiyaçlarının dışında bir ürün ise analiz aşamasına geri gönderir. Analizde düzeltilen alanlar yazılım ve test aşamalarından tekrar geçerek kullanıcıya gelir. Ekranda olan hatalar için kullanıcı ürünü yazılıma geri gönderir. Düzeltilen ürün tekrar test aşamasından geçerek kullanıcıya gelir. Sorun bulunmayan ürünü işbirimi proje sorumlusu ve işbirimi proje yöneticisi onaylar.

Tüm aşamaları tamamlanan ürünün 1 gün içerisinde yazılımcılar tarafından kodları UAT ortamından PROD ortamına alınır ve devreye alma işlemi tamamlanır. Devreye alımdan sonra rastlanılan hatalar gün içerinde çözülüp acil çıkışlar şeklinde iyileştirmeler yapılır.

Tablo 11. Yazılım Projesindeki Görevler ve Detayları

Görev İsmi	Görev Detayı
Bütçeleme	Önceki projelerle benzerlik inceleme
	Bütçe raporu hazırlama
Bütçe Onayı Alma ve Sponsor Atama	Bütçe raporu inceleme ve onaylama
	Sponsor atanması
Gereksinim Analizi Hazırlama	Gereksinimlerin detaylandırılması
	İş gereksinimleri belgesinin hazırlanması
	Gereksinim analiz dokümanı hazırlanması
	Gereksinim takip matrisi hazırlanması
	Önceki projelerle benzerlik inceleme
	Sistemler arası bütünleşme analizi
Tasarlama	Kullanıcı sistem tasarımı
	Önceki projelere benzerliklerin tasarımı
	Mimari Kontrat dokümanı hazırlanması
	Fiziksel altyapı tasarım dokümanı hazırlanması
	İş analizi ve tasarım gözden geçirmesi
Kodlama	Altyapı çalışmaları, donanım temini
	Kurulumların yapılması
	Kod geliştirme
	Ekran geliştirme
	Data sağlama
	Web Servis geliştirme
	Güvenlik testleri
	Birim testleri ve birim entegrasyon testleri
Test Etme	Test planı hazırlanması
	Test senaryoları hazırlanması
	Sürüm planı ve işletime alma planı hazırlanması
	Ürün tanım dokümanı hazırlanması
	Teste hazır olma gözden geçirmesi
	Test sonuç raporu hazırlanması
	Kabul testi tutanağı hazırlanması
Kullanıcı Test Etme	Kullanıcı kabul testleri
Kullanıcı Onaylama	Projenin son kullanıcı tarafından onayı
Devreye Alma	Kullanım kılavuzu hazırlanması
	Sürüm doğrulama kriterleri ve geri dönüş planı hazırlanması
	Üretime hazır olma gözden geçirmesi

Proje başlangıç ve bitiş dönemlerinde Şekil 39'daki gibi yol izlenir:



Şekil 39. Projede Akış Hayat Döngüsü

Kaynak: Ekho-Tec, (2017)

6.1.3. Mevcut Durum Değer Akış Haritasının Hazırlanması

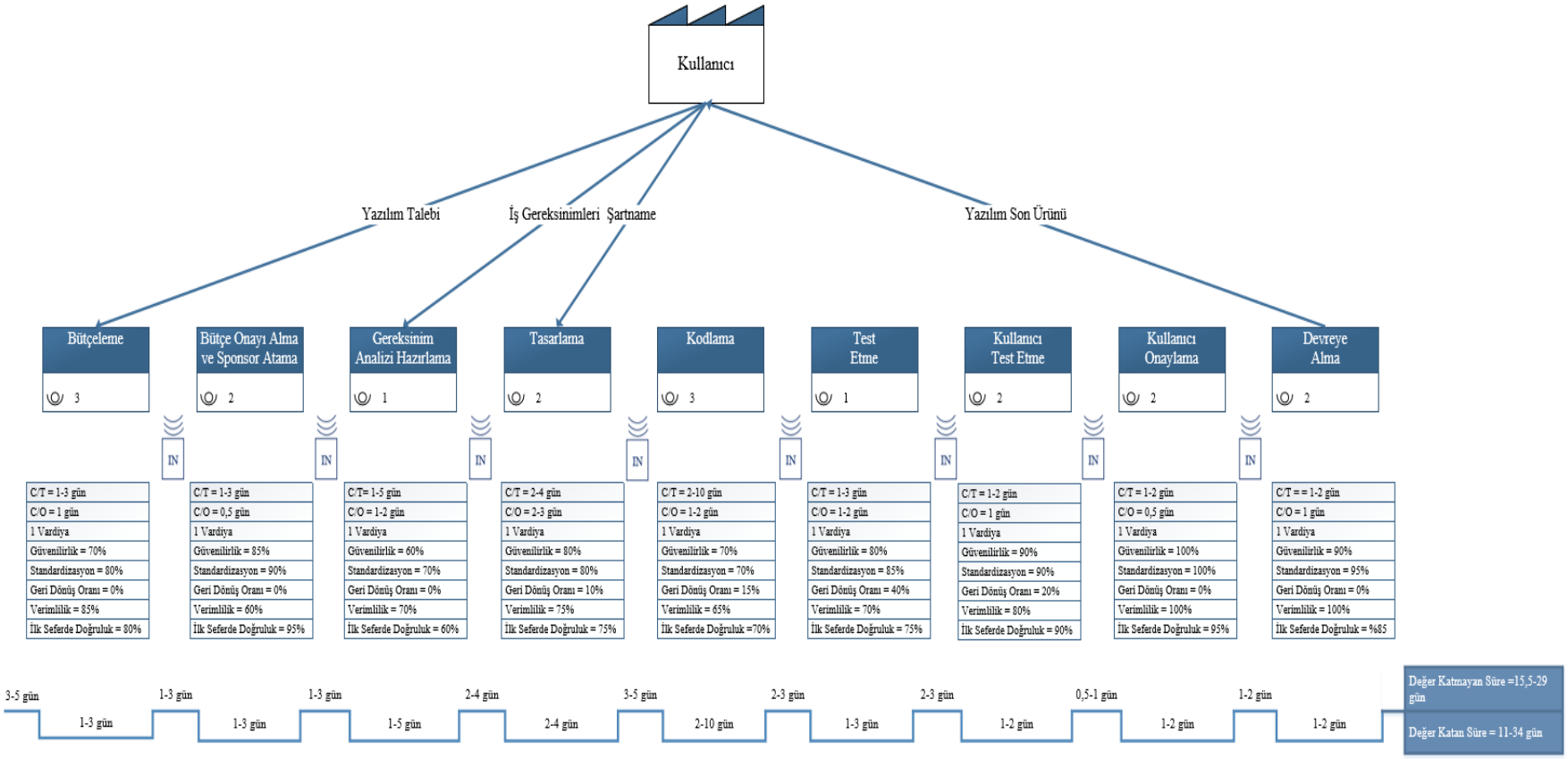
Projede akış yaratmak ve ürünün nasıl çalıştırılması gerektiğini detaylandırmak için değer akış haritalama yöntemi uygulanmıştır ve Mevcut Durum Değer Akış Haritası oluşturulmuştur. Mevcut duruma ait değer akış haritası Şekil 40'da gösterilmektedir.

Firmanın aldığı talepler büyüklüklere göre değişmektedir. Mevcut duruma ait değer akış haritasıda süreler taleplerin büyüklüklerine göre oluşturulmuştur. Kullanıcılar istek ve ihtiyaçları aylık belli periyodlarda BT departmanına bildirirler ve yapılması uygun görülen talepler işleme alınıp bütçe çalışmalarına başlanır.

Mevcut Durum haritasında, değer akış analizi yapılırken; gereksinim analizi hazırlama ve tasarlama aşamalarında kullanıcıya danışılmıştır. Süreçte 9 aşama bulunmaktadır ve şelale modelindeki gibi akış halinde hatlar tamamlanmıştır. Firmada tek vardiya sistemi uygulanmaktadır.

Mevcut Durum haritasına göre aşamalar arası toplam 15,5-29 gün aralığında bekleme süreleri bulunmaktadır.

Önceki projelerde detaylı hazırlanmayan bütçe planlama raporları, projelerin birikmesine ve bekleme sürelerinin oluşmasına sebep olmuştur. Bütçe onay sürecinde onaylayıcı olmadığı için yerine onay verecek vekaleti olmaması bekleme süresinin uzamasına neden olmuştur.



Şekil 40. Mevcut Durum Değer Akış Haritası

Talepler **Bütçeleme** aşamasına gelmeden önce büyüklüklerine göre 3-5 gün sırada beklemektedir. Müşteri istekleri, uygulanabilirlik, donanım yeterlilikleri, süre ve maliyet konuları detaylı şekilde değerlendirilmiştir, tüm konular açıklığa kavuşturulduktan sonra sözleşme hazırlanmıştır. Bu aşamada değerlendirilmeyen ya da gözden kaçan hususlar gereksiz masraflara, zaman kaybına hatta projenin yarım bırakılmasına neden olmuştur. Bütçeleme aşamasında projenin başlangıcından sonuçlandırılmasına kadar olan sürede izlenecek yol ve yöntemler ile donanım maliyeti, çalışan iş bölümü, proje yöneticisinin öngörü ve seçimleri, proje adımlarının sürece dağıtılması gibi unsurlar değerlendirilmiştir. Öncelikli olan talepler bütçeleme aşamasına girmiştir ve firma eski projeleri baz alarak bütçe araştırması yapılmıştır. Genellikle 1-3 gün sürede aşama tamamlanır. Yarım günlük hazırlama süreci bulunmaktadır. İşlem süresi hazırlıkla birlikte 1 gün sürmektedir. Fakat önceki projelerin dokümanlarının bulunmasının zaman alması, anlaşılamayan talepler, bütçeleme ve raporlama çalışanının izinde olması durumlarından 1-3 gün aralığında süre uzamıştır.

Bütçe raporu hazırlanan talepler için **Bütçe Onayı Alma ve Sponsor Atama** aşamasına geçilmiştir. Hazırlanan bütçe raporu onay için üst yönetime sunulmuştur. Bütçe raporunun ve talep dokümanının anlaşılamaması veya detaylı olmaması, üst yönetimin izinde veya toplantı amacıyla şirkette bulunmaması onay süresinin 1-3 gün aralığında olmasına neden olmuştur. Proje kapsamında üretilecek proje yönetim çıktıları proje yönetim aracına HP PPM (Hewlett Packard, versiyon: 9.30.0001) programına yüklenerek konfigürasyon kontrolüne alınmıştır. Diğer süreçlerle ilgili konfigürasyon birimleri için konfigürasyon kuralları ilgili süreç tanımlarında yapılmıştır. Talepler kapsamında standart konfigürasyon birimleri oluşturulmuştur. Sponsor ataması tamamlanmış talep için Tablo 12'deki gibi proje yöneticisi tarafından proje planı oluşturulmuştur.

Tablo 12. Yazılım Proje Planı Örneği

	Aşamalar	Teslimat	Tarih	Teslimat Sorumlusu	Yöntem	Kabul Sorumlusu	Kabul Kriterleri
1	Proje Kapsamını Belirlenmesi	Proje Başlatma Formu ve Proje Kapsam Dokümanı	May.16	Proje Yöneticisi	Dokümantasyon	Sponsor	Onay
2	Proje Yönetim Planının Hazırlanması	Proje Yönetim Planı	May.16	Proje Yöneticisi	Dokümantasyon	Sponsor	Onay
3	Talep detaylandırılması	Talep Detay Dokümanı	May.16	İş Birimi Proje Sorumlusu	Veri	Kullanıcı	Sözleşmeye uygun kapsam ve formatta veri
4	Sunucu ve veritabanı hazırlanması	Sunucu ve veritabanı	May.16	Bilgi İşlem Sorumlusu	Veri	Teknik Mimar	Kurulum
5	Gereksinim Analiz Dokümanı Hazırlanması	Gereksinim Analiz Dokümanı	May.16	İş Analizi Sorumlusu	Dokümantasyon	Proje Yöneticisi	Onay
6	Mimari Kontrat Dokümanı Hazırlanması	Mimari Kontrat Dokümanı	Haz.16	Teknik Mimar	Dokümantasyon	Proje Yöneticisi	Onay
7	Fiziksel Altyapı Tasarım Dokümanı Hazırlanması	Fiziksel Altyapı Tasarım Dokümanı	Haz.16	Altyapı Hizmet Uzmanı	Dokümantasyon	Proje Yöneticisi	Onay
8	İş Analizi ve Tasarım Gözden Geçirmesi	İA veTGG Raporu	Haz.16	Kalite Güvence Sorumlusu	E-Posta	Proje Yöneticisi	%70 gözden geçirme başarısı
9	Web Servis Geliştirme	Servis	Haz.16	Yazılım Geliştirme Sorumlusu	Uygulama	Proje Yöneticisi	Test kabulu
10	Ekran Geliştirme	Ekran, Akış	Haz.16	Yazılım Geliştirme Sorumlusu	Uygulama	Proje Yöneticisi	Test kabulu
11	Birim Testleri ve Birim Entegrasyon Testleri	Birim Test Sonuç Raporu	Haz.16	Yazılım Geliştirme Sorumlusu	Dokümantasyon	Proje Yöneticisi	Onay
12	Test Senaryoları Hazırlanması	Test Senaryoları	Haz.16	Test Sorumlusu	Dokümantasyon	Proje Yöneticisi	Onay
13	Test Planı Hazırlanması	Test Planı	Haz.16	Test Sorumlusu	Dokümantasyon	Sponsor	Onay
14	Sürüm Planı Hazırlanması	Sürüm Planı	Haz.16	Yazılım Geliştirme Sorumlusu	Dokümantasyon	Proje Yöneticisi	Onay
15	Teste Hazır Olma Gözden Geçirmesi	THOG Raporu	Haz.16	Kalite Güvence Sorumlusu	E-Posta	Proje Yöneticisi	%70 gözden geçirme başarısı
16	Test Sonuç Raporu Hazırlanması	Test Sonuç Raporu	Haz.16	Test Sorumlusu	Dokümantasyon	Proje Yöneticisi	Onay
17	Kabul Testi Tutanağı Hazırlanması	Kabul Testi Tutanağı	Haz.16	Test Sorumlusu, İş Birimi Proje	Dokümantasyon	Sponsor, Kullanıcı	Onay
18	Kullanım Kılavuzu Hazırlanması	Kullanım Kılavuzu	Haz.16	İş Birimi Proje Sorumlusu	Dokümantasyon	Sponsor, Kullanıcı	Onay
19	Üretime Hazır Olma Gözden Geçirmesi	UHOG Raporu	Haz.16	Kalite Güvence Sorumlusu	E-Posta	Proje Yöneticisi	%70 gözden geçirme başarısı
20	Proje Kapanışı	Proje Kapanış Raporu	Haz.16	Proje Yöneticisi	Dokümantasyon	Sponsor	Onay

Gereksinim Analizi Hazırlama aşamasına gelen talepler iş analistlerine paylaştırılmıştır. İş gereksinimlerini detaylandırmak amacıyla kullanıcı da bu aşamada yer almıştır. Talepler 1-5 günde tamamlanmaktadır. Kullanıcı ve iş analistinin toplantı zamanlarının verimli ayarlanamaması, kullanıcı isteklerinin anlaşılamaması bekleme süresini oluşturmuştur.

Gereksinim analiz dokümanı oluşturulan talepler **Tasarlama** aşamasına geçilmiştir ve Teknik Mimara iletilmiştir. Sistemin çalışmasıyla ilgili işlev ve görevler donanım, yazılım ve kullanıcılara dağıtılmıştır. Teknik mimar tarafından donanım için tanımlamalar yapılmıştır ve özel donanımlar tasarlanıp geliştirilmiştir.

Yazılım tasarım sürecinde aşağıdaki tasarımlar gerçekleştirilmiştir;

- Mimari tasarım
- Arayüz tasarımı
- Bileşen tasarımı
- Veri yapısı tasarımı
- Algoritma tasarımı

İşlem süresi talebin büyüklüğüne göre 2-4 gün olabilmektedir. Kullanılacak donanımlara göre yazılım ihtiyaçları belirlenmiş, çalışma ortamı ve veritabanı modellenmiş, kütüphaneler ve web servisler oluşturulmuş, kullanıcı ara yüzü tasarlanmış, sistem ilişkisel veri yapısı ve sistem altyapı ilişkisel yapısı oluşturulmuştur. Güvenlik açısından kriterler belirlenmiş ve kullanılacak ağ kontrolleri sağlanmıştır. Uygulama teknik detaylarıyla ilgili doküman oluşturulmuştur.

Kodlama aşamasında teknik mimar tarafından oluşturulan doküman yazılımcılara gönderilmiş, oluşturulan teknik detaylara göre kodlama yapılmaya başlanmış, hatalar giderilmiş ve modüller birleştirilerek yazılım oluşturulmuştur. Geliştirilen yazılım öğeleri, üretilen veya hazır olarak alınan donanım öğeleriyle tümleştirilmiştir. Tümleştirme işlemi karmaşık donanım yapılarının birlikte çalıştığı bazı büyük taleplerde oldukça dikkatli yapılması gereklidir. Bu aşama için üretilen prototipler üzerinde birim testler yapılmış, hata oranları incelenmiş, gerekli görülen durumlarda yazılımsal ve donanımsal düzenlemeler gerçekleştirilmiştir. Kodlama Java yazılım dili kullanılarak, Test ortamına yapılmıştır. Tümleştirmesi tamamlanan öğeler bir bütün haline getirilip sistemin asıl çalışacağı ortamda son bir test işlemine gönderilmiştir. Bu süreç aynı zamanda sistemin doğrulanmasını ve sağlamasını içermektedir. Kodlama aşamasında işlem süresi talep büyüklüğüne göre ortalama 2-10 gün sürmektedir. Bu aşamada 3 yazılımcı görev almıştır. Tasarımdan 3-5 gün geç gelen talepler birikmiştir, teknik dokümanlar detaylı olmadığı için talepler tasarıma geri dönmüştür ve eksik kısımlar tekrar tasarlanmışıtr. Sistemsel sıkıntılar, ağ problemleri ya da güvenlik açıkları bekleme süresinin artmasına neden olmuştur.

Kodlama aşaması biten talepler **Test Etme** aşamasına gönderilmiştir. Test Mühendisi talebe göre senaryolar oluşturmuş ve bu senaryoları firmada kullanılan HP ALM programına yüklemiştir. Oluşturduğu adımların test ortamında bu program üzerinden koşumunu sağlamıştır, bulduğu hataları program üzerinden yazılımcıya yönlendirmiş ve takibini buradan sağlamıştır. Günlük raporlar bu program üzerinden çıkarılmış ve takip için yöneticilere gönderilmiştir. Sistem arızaları, bulunan hataların büyüklüğü, ağ ortamının çalışmaması test sürecinde aksamalara neden olmuştur. 2-3 gün bekleme süresi oluşmuştur.. Testlerdeki hatalar çözümlendikten sonra kullanıcı testi için kodlar yazılımcı tarafından UAT ortamına aktarılmıştır.

Kullanıcı Test Etme aşamasında, kullanıcılar UAT ortamında teste başlamışlardır. Bunun içerisinde sistemin kullanımından sorumlu kişilere eğitim verilmesi ve ilk kullanım sonunda ortaya çıkan hataların giderilmesi gelmektedir. İşlem süreleri 1-2 gün olarak belirlenmiştir. Testte kullanılacak verilerin hazırlanması yarım gün sürmektedir. Bulunan hatalar yine HP ALM üzerinden yazılımcılara gönderilmiştir. Çözülen hatalar önce testçiye sonra kullanıcıya test tekrarı olarak gönderilmiştir.

Hataların çözümü ve testlerin tamamlanmasından sonra **Kullanıcı Onaylama** aşamasına geçilmiştir. Teslim zamanına yetişemeyecek büyüklükte hatalar bir sonraki sürümde çözülmek üzere listeye alınmıştır. Geçerli kriterleri tamamlayan talep süreçleri kullanıcı tarafından onaylanmıştır.

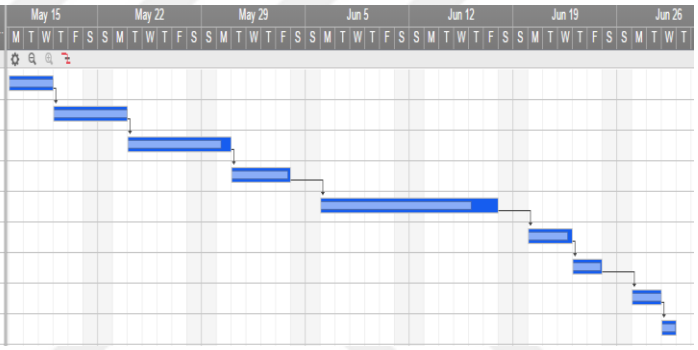
Devreye Alma aşamasında onaylanan taleplerin kodları yazılımcılar tarafından UAT ortamından PROD ortamına yani canlı ortama aktarılmıştır. Her ay sürüm çıkışı yapılmaktadır. Devreye alım 1 gün sürmektedir ama sistemsel sıkıntılardan ya da kodların yanlış aktarımından kaynaklı süre 2 güne çıkabilmektedir.

6.1.4. Mevcut Durum Değer Akış Haritasının Analizi ve Gelecek Durum Değer Akış Haritası Tasarımı

Harita işlem süreleri ve hazırlık süreleri açısından değerlendirilmiştir. Mevcut durum değer akış haritasına bakıldığında en yüksek işlem süresine sahip hattın kodlama olduğu görülmüştür. En uzun hazırlık sürelerinin de kodlama ve test etme olduğu belirlenmiştir. Bu analizler sonucunda kodlama ve test etme hatlarında hazırlık sürelerini düşürmeye yönelik çalışmalar yapılması gerekliliğini ortaya koymaktadır.

Tablo 13. Mevcut Durum İçin Oluşturulan Gantt Tablosu

Task Name	Duration	Start	Finish	% Complete	Status	Prede...
Bütçeleme	3d	05/16/16	05/18/16	100%	Complete	
Bütçe Onay	3d	05/19/16	05/23/16	100%	Complete	1
Analiz Etme	5d	05/24/16	05/30/16	90%	Complete	2
Tasarlama	4d	05/31/16	06/03/16	95%	Complete	3
Kodlama	10d	06/06/16	06/17/16	85%	Complete	4
Test Etme	3d	06/20/16	06/22/16	90%	Complete	5
Kullanıcı Test Etme	2d	06/23/16	06/24/16	95%	Complete	6
Kullanıcı Onay	2d	06/27/16	06/28/16	100%	Complete	7
Devreye Alma	1d	06/29/16	06/29/16	95%	Complete	8



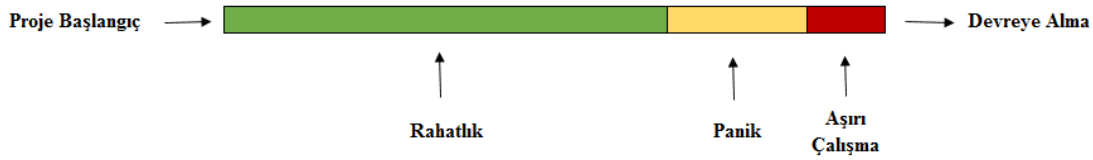
Tablo 13'teki Gantt tablosuna göre yeterli tamamlanma oranları sağlandığında bir projenin 33 günde tamamlandığı görülmektedir. Bu durum diğer taleplerin birikmesine ve ekstra maliyete sebep olmaktadır. Mevcut Durum haritasında 11-34 gün aralığında talebin tamamlandığı görülmektedir. Gantt tablosunda bir talebin ortama ne kadar günde tamamlandığı planlanmıştır. Kriter oranları da Tablo 14'teki gibidir:

Tablo 14. Mevcut Durumda Kriterlerin Aldığı Değerler

	Bütçeleme	Bütçe Onayı Alma ve Sponsor Atama	Gereksinim Analizi Hazırlama	Tasarlama	Kodlama	Test Etme	Kullanıcı Test Etme	Kullanıcı Onaylama	Devreye Alma
Güvenilirlik	70%	85%	60%	80%	70%	80%	90%	100%	90%
Standardizasyon	80%	90%	70%	80%	70%	85%	90%	100%	95%
Geri Dönüş Oranı	0%	0%	0%	10%	15%	40%	20%	0%	0%
Verimlilik (Çalışma Süreleri)	85%	60%	70%	75%	65%	70%	80%	100%	100%
İlk Seferde Doğruluk	80%	95%	60%	75%	70%	75%	90%	95%	85%

Firmanın kullandığı performans ölçümleme programından alınan rapora göre kriterlerle ilgili yazılım geliştirme modelinde iyileştime çalışmaları yapılması gerekmektedir. Özellikle Gereksinim Analizi Hazırlama güvenilirlik oranı oldukça düşüktür. Analiz edilirken önceki projeler araştırılmalıdır. Müşteriden istek ve ihtiyaçlar detaylı bir şekilde öğrenilmeli ve doküman uygun bir dille oluşturulmalıdır. Detaylı oluşturulmadığı sürece diğer aşamalar bu analize bağlı olduğundan projenin aksamasına neden olur. Dokümanda aşırı ayrıntıya girmek de hataya sebep olur. Çünkü Yazılım aşamasında yazılımcılar analizi çok detaylı okumamaktadırlar.

Planlamada yapılan hatalar projenin tüm seyrini etkilemektedir. Bu hatalar talebi basit görmek, büyüklüğünü tahmin edememek, önceki başarıları yanlış değerlendirmek ve dış baskılara boyun eğmek neticesinde oluşmaktadır. Analiz tamamlanmadan yazılıma başlanması projenin istenen kaliteye ulaşmasına engel olmaktadır. Önceki benzer projelerin analizinde önceki planlar aynen kullanıldığı için güvenilirlik oranı %60 olarak tespit edilmiştir. Genellikle proje yaşam süreci Şekil 41'deki gibi verimsiz olmaktadır. Projelerde sürekli yaşanan gecikmelerden ve aksaklıklardan dolayı birimler arasında da sürtüşmeler meydana gelmektedir. Örneğin bu projede yazılım geliştirme ekibi iş birimlerini isteklerini net oluşturmamaktan ve sürekli farklı talepler öne sürmelerinden dolayı suçlamaktadır. İş biriminin ek istekleri projenin sapmasına yol açmaktadır. Ayrıca proje yönetim birimine de sürekli baskı yapmalarından dolayı şikayetçilerdir. İş birimi ise yazılım geliştirme ekibini yavaşlıkla suçlamaktadır. Her gün yaşanan birimler arasındaki iletişimsizlik verimsizliği arttırmaktadır.



Şekil 41. Talep Teslim Süreci

Ofis ortamlarının uzaklığı da proje sürecini etkilemektedir. İş birimi ve yazılım geliştirme biriminin uzaklığı iletişimsizliği artıran unsurlardan biridir. Firmada işbirimi 1. katta, yazılım geliştirme ekibi 6. katta, proje yönetim birimi 4. katta bulunmaktadır. Bu durum toplantılara gecikme veya gelmeme durumuna neden olmaktadır. Bir diğer durumda da planlama, bütçeleme ve analiz aşamalarında önceki projeler baz alındığından döküman ihtiyacı artmaktadır. Dolaplardaki dökümanların karmaşıklığı istenen evrakları bulmayı zorlaştırmaktadır. Dolaplara sığmayan evrakların masa üstlerinde biriktirilmesi oldukça çirkin bir görüntü oluşturmaktadır. Ayrıca bilgisayarda ortak bir alan olmayışı da büyük bir zaman kaybına neden olmaktadır. Bu durumu düzeltmek adına 5S uygulamasına geçilmiştir. Seiri (Ayıklama) aşamasında gerekli ve gereksiz nesneler ayıklanmıştır. Gereksiz nesneler kırmızı etiketlerle belirlenip, kırmızı alanlara taşınmıştır. Dosyalar dolaplara sıralı ve numaralandırılmış bir şekilde yerleştirilmiştir. Son olarak da bu değişim firmada standartlaştırılıp rutin hale getirilmesi sağlanmıştır.

Projede çalışanların eğitim seviyeleri de süreci etkilemektedir. Yazılım geliştirme birimindekilerin bilgisayar, matematik veya yazılım mühendisliği, iş biriminin iktisadi ve idari bilimlerden, proje yönetim biriminin ise işletme veya endüstri mühendisliği bölümlerinden mezun olması gerekmektedir. Bu projede:

- Yazılımcılar Bilgisayar Mühendisliği,
- İş birimi İşletme,
- Proje yöneticisi Endüstri Mühendisliği eğitimine sahiptir.

Projelerin aksaması arttıkça çözüm yolu aranmaya başlanmıştır ve Yalın ilkelerini desteklediği için Çevik dönüşüm yazılım metodolojilerinden **SCRUM** manifestosunun denenmesine karar verilmiştir. Çevik dönüşüm firmada ilk defa

kullanılacağı için bilişim departmanındaki çalışanlara bu konuda uzman bir firmadan SCRUM eğitimi aldırılmıştır. Bu proje de pilot proje olarak belirlenmiştir. Proje için aşağıdaki gibi SCRUM Takımı oluşturulmuştur:

- ***SCRUM Master*** : 1 Çalışan,
- ***Product Owner (Ürün Sahibi)*** : 1 Çalışan,
- ***SCRUM Geliştirme Takımı (Development Team)*** : 5 Çalışan

Proje süreci aşağıdaki SCRUM adımlarının uygulanmasıyla ilerlemiştir:

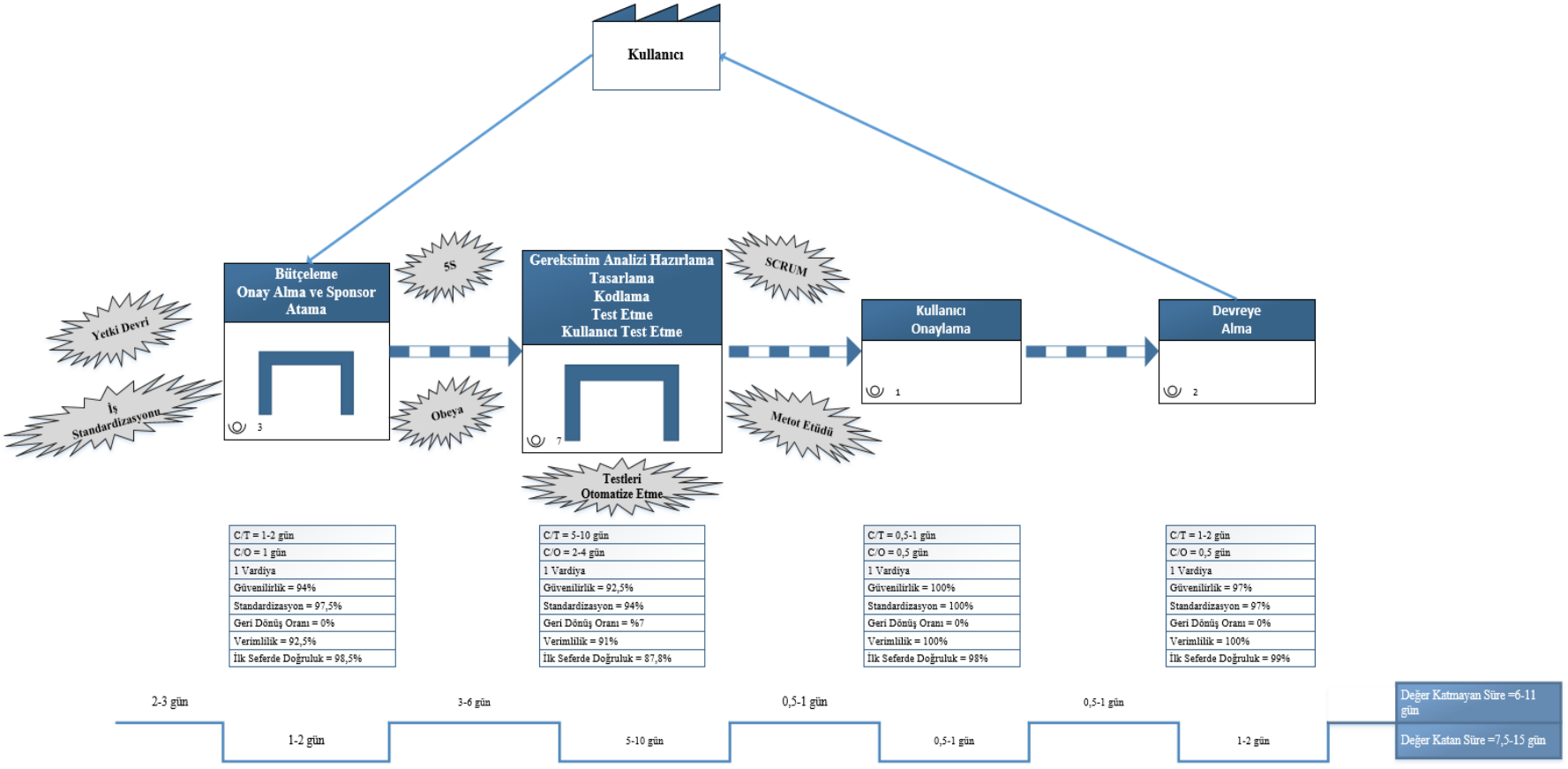
- Ürün Sahibi geliştirilmesi gereken tüm fonksiyonları önceliklendirerek ürün kaydını (Product Backlog) hazırlamıştır.
- Takım belirlendikten sonra ve "Sprint Planning" denilen en fazla dört hafta sürecek olan küçük döngülerle "Sprint" ile işe başlanmıştır. Her döngü için sprint kaydı (Sprint Backlog) oluşturulup, sprint boyunca geliştirme yapılmıştır.
- Takım bu "Sprint" süresince her gün Scrum Master liderliğinde bir araya gelip (Daily Scrum Meeting), en fazla 15 dakika içerisinde her takım üyesi kendi ilerlemesini anlatmışlardır. Sprint süresince kalan gereksinimler ve geçen zaman grafiği güncellenmiştir.
- Sprint bittiği zaman bir "Sprint review" raporu çıkarılıp, sprint süresince ortaya çıkan problemler ortadan kaldırılarak bir sonraki sprinte daha rahat başlanmıştır.
- Sprint sonlarında tamamlanan her parça kullanıcıya gösterilmiştir.

Yapılacak işler küçük iş parçacıklarına bölünüp, SCRUM tahtasında belirtilmiştir. Ekip bu şekilde çalışırken kendi aralarında yapıcı fikir paylaşımında bulunarak alternatif çözümler yaratmışlardır. Bu projede zaman tasarrufu sağlamak amacıyla regresyon otomatize edilmiştir.

Gelecek Durum Haritası oluşumunda taleplerin karşılanma sürelerinin kısaltılması amacıyla Çevik yazılım metodolojilerinden SCRUM manifestosu benimsenmiştir. Yalın teknikler kullanılarak hatalar azaltılıp, iyileştirmeler yapılmıştır. Kısa süreli kilometre taşı veya kontrol noktaları tanımlamak, yapılanlarla

planın kısa aralıklarla karşılaştırılması sağlanarak, plandan sapılmasına engel olunmuştur.

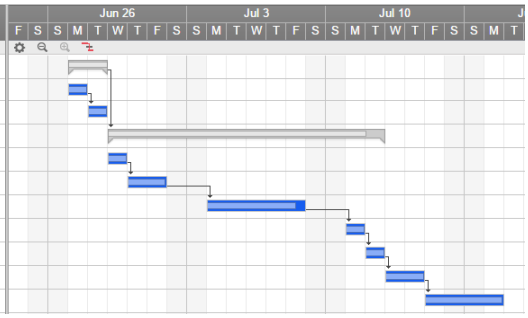
Şekil 42'de görülen Gelecek Durum haritası analiz edildiğinde Bütçe ve yazılım geliştirme aşamalarının hücreselleştirildiği görülmektedir. Bütçeleme ve onay süreci hücreselleştirilerek Mevcut Durum haritasında görülen 1-3 gün bekleme süresinin ortadan kalktığı görülmektedir. Ayrıca Gereksinim Analiz Hazırlama, Tasarlama, Kodlama, Test Etme ve Kullanıcı Test Etme aşamaları da hücreselleştirilerek 9-15 olan bekleme süresi de sıfırlanmıştır. İki Hücre arasında süpermarket sistemi kurulmasına karar verilmiş ve bekleme süresi 6-11 güne düşürülmüştür.



Şekil 42. Gelecek Durum Değer Akış Haritası

Tablo 15. Gelecek Durum İçin Oluşturulan Gantt Tablosu

Task Name	Duration	Start	Finish	Predecess...	% Complete	Status
Bütçeleme	2d	06/27/16	06/28/16		100%	
Bütçe Oluşturma	1d	06/27/16	06/27/16		100%	
Bütçe Onay	1d	06/28/16	06/28/16	2	100%	
Yazılım Geliştirme	10d	06/29/16	07/12/16	1	93%	
Analiz Etme	1d	06/29/16	06/29/16		95%	
Tasarlama	2d	06/30/16	07/01/16	5	95%	
Kodlama	5d	07/04/16	07/08/16	6	90%	
Test Etme	1d	07/11/16	07/11/16	7	95%	
Kullanıcı Test Etme	1d	07/12/16	07/12/16	8	95%	
Kullanıcı Onay	2d	07/13/16	07/14/16	9	100%	
Devreye Alma	2d	07/15/16	07/18/16	10	98%	



Gantt tablosuna göre kabul görülen tamamlanma oranları tutturulduğunda ilk uygulamada projenin 16 günde tamamlandığı görülmektedir. Hücreler içerisinde özellikle yazılım geliştirme hücresinde Yalın felsefe benimsenmiştir ve aşağıdaki ilkeler amaçlanmıştır:

- **İsrafı Giderme:** Sistemin en önemli ilkesi kabul edilmiştir. Müşterinin talebini öncelikli tutmak ve bu talebi daha etkili kullanabilmek hedeflenmiştir. Yazılım geliştirme aşamasında müşterinin talep etmediği, zorunlu olmayan gereksinimler prosedürden çıkarılmıştır.
- **Bütünü Optimize Etme:** Bu ilke yazılım geliştiricilere sistemi bir bütün olarak ele almalarını, araştırmalarını ve anlamalarını önerir. Yazılım geliştirme ekibi sadece test aşamasında iyileştirmeye gidilmesi değil, tüm yazılım geliştirme sürecinde iyileştirilmeye gidilmeyi amaç edinmiştir.
- **Kaliteyle Geliştirme:** Yazılım geliştirmenin farklı aşamalarında kullanılan gözden geçirme aktiviteleri, artık kalite standartlarına göre yapılmaya başlanmıştır. Çevik geliştirmede kullanılan Test Güdümlü Geliştirme ve Eşli Programlama teknikleri kullanılmaya başlanmıştır.
- **Devamlı Öğrenme:** Teknolojinin devamlı gelişmesi, yazılım geliştirme ekibinin bilgilerini sürekli güncellemesi gereklidir. Bu yüzden yazılım geliştirme ekibi sürekli yeni bilgiler öğrenmeyi hedeflemişlerdir.
- **Hızlı Teslim:** Yazılım geliştirme ekibi hızlı bir şekilde ürün geliştirmeyi ve müşteriye hızlı teslimi amaçlamışlardır.
- **Takımı Güçlendirme:** Ekip içerisinde yazılım geliştiriciler birbirlerine güvenmeyi, saygı duymayı ve yardımlaşmayı benimsemişlerdir.

- **Zamanında Karar Verme:** Hızlı karar vermek hataların artmasına neden olduğu için biraz daha uzun sürede düşünülerek ve ekip içerisinde fikir alışverişi yapılarak kararların alınması amaçlanmıştır.

6.1.5. Değer Akış Planının Oluşturulması ve Uygulamanın Değerlendirilmesi

Çalışmanın ana fikri olan proje yönetiminde süreç iyileştirmede değer akış haritalama yöntemi kullanılmıştır. Çevik yazılım geliştirme modellerinden SCRUM ve Yalın Teknikler benimsenerek proje üzerinde iyileştirmeler yapılmıştır. Tablo 16'da iyileştirilmesi gereken alanlar, süreler ve sorumlu yöneticiler açıklanmıştır. Üretimde yıllık olarak oluşturulan değer akış planı, bilişimde talepler kullanıcıdan aylık geldiği ve 1 ay içerisinde çözümlenmesi gerektiğinden tablo aylık plan olarak hazırlanmıştır.

Tarih:	10.05.2016	AYLIK DEĞER AKIŞ PLANI	İmzalar			
Alan Yöneticisi:	Ecrin BAŞARIR		Genel Müdür Yardımcısı	Bütçe Departmanı	İş Birimi	Proje Yöneticisi
Değer Akış Yöneticisi:	Burcu TANRIKUT ÇERKEZOĞLU					

Ürün Ailesi İş Amacı	Değer Akış Çevrimi	Değer Akışının Amacı	HEDEF (Ölçülenebilir)	GÜNLÜK PROGRAM																				Görevli Kişiler	Bölümler	Gözden Geçirme Planı	
				1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20			Gözden Geçiren	Tarih
Yazılım projelerinde kaliteli ürün ve hızlı teslimat sağlanması için süreci iyileştirmek.	Bütçeleme Onay Alma Ve Sponsor Atama	Hazırlık sürelerinin düşürülmesi	C/T ≤ 2 gün																					Bütçe ve Raporlama Uzmanı	Bütçe ve Raporlama	Bütçe ve Raporlama Müdürü	16.05.2016
		Onaylama sürecindeki bekleme süresinin azaltılması; Yetki Devri	C/O ≤ 1 gün																					Genel Müdür Yardımcısı	Üst Yönetim	Genel Müdür Yardımcısı	16.05.2016
		İş Standardizasyonunu sağlamak	97,50%																					Bütçe ve Raporlama Uzmanı	Bütçe ve Raporlama	Bütçe ve Raporlama Müdürü	16.05.2016
	Yazılım Geliştirme	İlk seferde doğruluk oranının arttırılması, SCRUM	87,80%																					Teknik Mimar, Yazılım Mühendisi	Bilgi Teknolojileri	Proje Yöneticisi	16.05.2016
		Ürün teslim süresinin düşürülmesi	C/T ≤ 10 gün																					Yazılım Mühendisi	Bilgi Teknolojileri	Proje Yöneticisi	16.05.2016
		Kaliteli ürün sunulması; Testlerin Otomatize Edilmesi	Güvenilirlik 92,5%																					Test Mühendisi	Bilgi Teknolojileri	Proje Yöneticisi	16.05.2016
		Ekip arasında etkili iletişimin arttırılması; 5S, Obeya, Metot Etüdü	Verimlilik 91%																					Proje Yöneticisi	Bilgi Teknolojileri	Proje Yöneticisi	16.05.2016
		Maliyetlerin düşürülmesi	55%																					Proje Yöneticisi	Bilgi Teknolojileri	Proje Yöneticisi	16.05.2016
	Kullanıcı Onaylama	Hazırlık sürelerinin düşürülmesi	C/T ≤ 1 gün																					İş Birimi Müdürü	İş Birimi	Proje Yöneticisi	16.05.2016
	Devreye Alma	Hazırlık sürelerinin düşürülmesi	C/T ≤ 2 gün																					Proje Yöneticisi	Bilgi Teknolojileri	Proje Yöneticisi	16.05.2016
		Sistemsel sıkıntıların minimuma düşürülmesi	İlk seferde doğruluk 99%																					Yazılım Mühendisi	Bilgi Teknolojileri	Proje Yöneticisi	16.05.2016

Tablo 16. Değer Akış Planı

Değer katan süreyi azaltmak için yalın yöntemler kullanılmıştır. 5S metodu ile çalışma ortamı uygun hale getirilen ofiste, projede zaman ve eğitim açısından metot etüdü yöntemi de kullanılmıştır. Mevcut günlük çalışma saatleri (8 saat) içinde bir saatlik iş kaybı proje sürecini etkileyeceğinde zaman etüdü yapılarak kayıp minimuma indirilmiştir. Çalışanın dikkatsizliği, geç gelmesi veya erken çıkması, yavaş tempoda çalışması, iş başında başka şeylerle ilgilenmesi zaman kaybı nedenlerindendir. Örneğin, yazılımcıların çok fazla ara vermesi süreci aksatmakta ve hatalı kod yazma oranını arttırmaktadır. 3 günde teste gönderilmesi gereken talep bu yüzden 5-6 günde teste gönderilmektedir. Test ekibinde işlerin birikmesi test mühendisinin tek bir işe odaklanmamasına neden olmakta ve bu durum hataların gözden kaçmasına sebep olmaktadır. Metot etüdü sonucunda:

- Çalışanın çalışma hızı yükseltilmiştir
- Daha iyi çalışma koşulları sağlanmıştır,
- Adil iş yükü dağılımı sağlanmıştır,
- Verimli çalışma koşulları geliştirilmiştir,
- Çalışanın gereksiz hareketleri ortadan kaldırılmıştır.

Metot etüdünde izlenilecek yol sistematik bir yapıya sahiptir. Öncelikle iş seçimi yapılmıştır. Gelen talepler öncelik derecelerine göre sıralanmıştır. Sıralanan bu talepler için planlama oluşturulmuştur ve kayıtlara geçirilmiştir. İş yükleri adil olarak ekip üyerine dağıtılmıştır. Çalışma için büyük bir toplantı odası ayarlanmıştır. Bu sayede ekip içinde etkili iletişim sağlanmıştır ve gereksiz hareketler ortadan kaldırılmıştır. Süreklilik sağlamak için standardizasyon sağlanmıştır.

Yalın perspektifinden bakıldığında çalışanların farklı odalarda bulunmaları verimlilik açısından proje sürecini olumsuz etkilemektedir. Çözüm olarak hücreselleştirilen yazılım geliştirme ekibi için proje takip ve değerlendirme amaçlı Obeya Odasinin kullanımına başlanmıştır. İletişim artiran Obeya Büyük Oda sistemi ile çalışanların her konuda eş zamanlı olarak görüşlerini bildirmeleri ve değerlendirmelerini paylaşabilmeleri ön plana çıkartılmıştır. Ekibin bir odada olması süreçlere odaklanılmasını sağlamıştır. Farklı alanlarda uzmanlaşmış çalışanlar aynı odada bulunmalarıyla sürekli birbirlerinden bir şeyler öğrenme eğiliminde olmuşlardır. Tüm proje verileri, çizimler ve grafikler odadaki duvarlara

yapıştırılmıştır. Bu nedenle Yalın ve Çevik Proje Yönetimi, tek bir odada ya da grup masada birlikte çalışan ekiplerin bulunmasını önermektedir. Teslimatı hızlandırmak, işin kalitesini artırmak, projenin maliyetini düşürmek ve karşılıklı öğrenmek ve sürekli iyileştirmeyi artırmak için Obeya tekniğinin uygulanması oldukça yardımcı olmuştur. Ekip üyeleri başka bir talep için başka bir odaya ya da grup masasına taşınmışlardır.

Tablo 17. Gelecek Durumda Kriterlerin Aldığı Değerler

	Bütçeleme	Bütçe Onayı Alma ve Sponsor Atama	Gereksinim Analizi Hazırlama	Tasarlama	Kodlama	Test Etme	Kullanıcı Test Etme	Kullanıcı Onaylama	Devreye Alma
Güvenilirlik	90%	98%	87%	95%	90%	95%	95%	100%	97%
Standardizasyon	95%	100%	95%	95%	86%	94%	95%	100%	97%
Geri Dönüş Oranı	0%	0%	0%	2%	8%	20%	5%	0%	0%
Verimlilik (Çalışma Süreleri)	95%	90%	95%	90%	85%	90%	95%	100%	100%
İlk Seferde Doğruluk	97%	100%	85%	90%	85%	87%	92%	98%	99%

Tablo 17'de görülen raporlama programından çekilen verilere bakıldığında SCRUM uygulanmasıyla kriter oranlarında büyük iyileşme görülmektedir. Çalışan sayısındaki fazlalıklarda ekipte iletişimsizliği arttırmaktaydı. Şelale modeli uygulanırken 18 çalışan varken, SCRUM modeliyle çalışan sayısı 13 kişiye düşürülmüştür. Bu 10 kişinin 7'si yazılım geliştirme ekibi, 3'ü bütçeleme ve onay ekibi olarak hücreselleştirilmiştir. Böylece ekip içinde iletişim daha da kuvvetlendirilerek yanlış anlaşılmalar giderilmiştir.

SCRUM, şeffaflık sağlayarak, ekip içinde iletişimi kuvvetlendirerek, gereksinimlerdeki hızlı değişikliklere ayak uydurarak ve proje teslim sürecine kadar olan sorunların çözülmesini hedeflemektedir.

Ekho-Tec firmasında gerçekleştirilen DAH uygulaması, mevcut durumda değer katan sürenin 11-34 gün olduğunu göstermiştir. Gelecek durum tasarımıyla bu süre 7,5-15 güne indirilerek % 45,7 oranında zamandan tasarruf olacağı belirlenmiştir. Güvenilirlik, Standardizasyon, Geri Dönüş Oranı, Verimlilik ve İlk

Seferde Doğruluk kriterlerinde yüksek oranda iyileşme görülmüştür. Kısa sürede kullanıcıya sunulan ürün geliştirmeleri ile kullanıcı memnuniyeti sağlanmıştır. Kullanıcı Çevik model sayesinde projenin her aşamasında bulunmuş ve bu sayede taleplerin anlaşılmasında sıkıntı yaşanmamıştır. Hücresel ekipleşme sayesinde de iş paylaşımı yapılarak talepler daha hızlı bir şekilde kullanıcıya sunulmuştur.

Tablo 18. Mevcut ve Gelecek Durum Kriterleri Arasındaki Oranlar

	Bütçeleme	Bütçe Onayı Alma ve Sponsor Atama	Gereksinim Analizi Hazırlama	Tasarlama	Kodlama	Test Etme	Kullanıcı Test Etme	Kullanıcı Onaylama	Devreye Alma
Güvenilirlik	20%	13%	27%	15%	20%	15%	5%	0%	7%
Standardizasyon	15%	10%	25%	15%	16%	9%	5%	0%	2%
Geri Dönüş Oranı	0%	0%	0%	-8%	-7%	-20%	-15%	0%	0%
Verimlilik (Çalışma Süreleri)	10%	30%	25%	15%	20%	20%	15%	0%	0%
İlk Seferde Doğruluk	17%	5%	25%	15%	15%	12%	2%	3%	14%

Tablo 18'de görüldüğü üzere en büyük artış oranı %30 oranla 'Bütçe Onayı Alma ve Sponsor Atama' aşamasında görülmektedir. İş standardizasyonu oluşturarak ve yetki devrinde iyileştirme yapılarak artış sağlanmıştır. Önceden projeler belli isimler tarafından onaylanan talepler, o yöneticiler izindeyken talep onayı beklemeye alınıyordu. Artık o kişiler izinde olduğu zaman başka yöneticilere yetki verilerek proje onay akışının aksaması engellenmiştir. Önceden iş onay prosedürleri detaylı dokümante edilmemekteydi. Artık onay sürecinde bütçe dokümantasyonu onay prosedürleri daha detaylı bir şekilde dokümante edilmektedir.

Bir diğer önemli artış olarak Şelale yöntemi uygulanırken Gereksinim Analizi Hazırlama güvenilirlik oranı %60 iken, Çevik yöntem ile bu oran %87'ye yükselmiştir, böylece %27'lik bir artış sağlanmıştır. Bu durumun en büyük sebebi işin her aşamasında iş biriminin de bulunmasından kaynaklıdır.

Hata geri dönüş oranlarında da önemli bir düşüş gözlemlenmiştir. İş parçaçıkları şeklinde her bir parça sunulduktan sonra test yapılması hatalara anında

müdahaleyi sağlamıştır bu da proje sürecini hızlandırmıştır. Obeya metodu kullanılarak analiz, tasarım, kodlama, test aşamaları hücresele döndürüldüğünden hücre içinde sürekli akış halindedir. SCRUM ekibinde hücre içindeki her çalışan bir diğerinin işini yapabilmektedir. Şelale modeldeki gibi çalışanların kesin bir görevi bulunmamaktadır. Bireysel değil ekip olarak hareket edilmekte ve paylaşımcı bir tutum izlenmektedir. Şelale modeldeki bekleme süreleri de bu sayede kalkmıştır. Hücresel çalışma, bekleme sürelerini azaltmada da etkili oluştur. Process'ler arası Şelale modelinde bekleme süresi 15,5-29 gün iken, Çevik modeli uygulandığında bekleme süresi 6-11 güne düşmüştür. Böylece talep birikmesi ortadan kalkmıştır.

Verimli çalışma sürelerinin oranlarında da artış görülmektedir. Yalın tekniklerden 5S ve metot etüdü uygulamalarının katkısından kaynaklanmaktadır. Obeya metodu büyük toplantı odasında çalışan ekip, masalara iş harici eşyalar koymamaktadır. Önceden istenilen evrak ve geçmiş projelere ait dokümanların bulunması zaman alırken artık her belgenin, dokümanın yeri bellidir ve vakit kaybı yaşanmamaktadır.

7. SONUÇ VE DEĞERLENDİRME

Küreselleşen ve rekabetin yoğunlaştığı dünyada kullanıcı memnuniyetini ve devamlılığını sağlamanın önemini tüm kurumlar kavramıştır. Yeni ürünler yaratmak için firmalarda çoğunlukla geleneksel yazılım geliştirme stratejileri kullanılmaktadır. Geleneksel yazılım geliştirme stratejileri, müşteri ihtiyaçlarındaki ani değişime cevap veremediği, projelerin fazla mesai harcanmasına ve belirlenen bütçeyi aşmasından dolayı teslim aşamasında sıkıntıya neden olmaktadır. Projeleri hızla kullanıcıya sunmak ve değişen gereksinimlere karşılık vermek amacıyla geleneksel yazılım geliştirme modellerinin yerini hızlı sunum, düşük maliyet, kalite ve yüksek üretkenliği kapsayan çevik yazılım geliştirme modelleri almaya başlamıştır.

Yazılım süreci belirsizliklerin çok olduğu aşamalardır. Yazılımdaki problemler yapısız ve düzensiz problemlerdir. Yazılım projelerindeki başarı oranlarını arttırabilmek için takım oyunu, önceliğe uygun sürekli teslimat, müşteri ile iş birlikteliği ve değişime adaptasyon olmak üzere dört önemli etken bulunmaktadır.

Bu çalışmada Bilgi Teknolojileri firmasında bütçeleme, gereksinim analizi hazırlama, tasarlama, kodlama, test etme aşamalarında oluşan bekleme süreleri değer akış haritasıyla belirlenmiş, yalın teknikler kullanılarak süreçte iyileştirmeler yapılmıştır.

Mevcut durum değer akış haritası analiziyle aşamalar arasında 15,5-29 gün bekleme süresinin oluştuğu ortaya çıkmıştır. Özellikle tasarlamadan kodlamaya, kodlamadan test etmeye geçiş aşamasında bekleme sürelerinin fazlalığı gözlemlenmiştir. Ekiplerin iletişimden uzak olması, anlaşılamayan müşteri istekleri, planlamalardaki hatalar bekleme süresinin artmasında başlıca etkenlerdir. Yalın tekniklerden 5S, Metot Etüdü, Obeya kullanılarak yazılım geliştirme sürecinde iyileştirmeler yapılmıştır. Pilot projeye yalın felsefeye yakınlığıyla bilinen çevik metodolojilerden SCRUM manifestosu uygulanmıştır. Bekleme sürelerinin fazla olduğu aşamalar hücreselleştirilerek iyileştirme sağlanmıştır. Değer katan süre önceden 11-34 gün iken, aşamalar hücreselleştirildikten sonra 7,5-15 güne

düşmüştür. Aynı şekilde değer katmayan süre 15,5-29 günden, 6-11 güne düşmüştür. Kullanıcı her aşamada bulunarak ürün isteklerine göre oluşturulmuş, ilk seferde doğruluk oranı %15 artmış, hata oranı %8 azaltılmış ve müşteri memnuniyeti sağlanmıştır.

Yalın teknikler sayesinde kriterler arası oranlarda artış görülmüştür. En büyük artış oranı %30 oranla 'Bütçe Onayı Alma ve Sponsor Atama' aşamasında görülmüştür. Gereksinim analizi güvenilirlik oranı %60 iken, yalın teknikler ile bu oran %87'ye yükselmiştir, böylece %27'lik bir artış sağlanmıştır.

Yalın felsefe ilkeleri benimsenerek üretimin ve kalitenin başarılı bir şekilde arttırılması hedeflenmiştir. Firmada bu ilkeler sayesinde müşteri talepleri daha başarılı olarak devreye alınmaya başlanmıştır.

Obeya Büyük Oda sistemi ile ekipler hücreselleştirilerek etkin iletişim sağlanmıştır. 5S metodu uygulanarak gerekli ve gereksiz nesneler ayıklanmıştır ve bu değişimin firmada standartlaştırılıp rutin hale getirilmesi sağlanmıştır.

Projede zaman ve eğitim açısından metot etüdü yöntemi kullanılmıştır. Mevcut günlük çalışma saatleri (8 saat) içinde bir saatlik iş kaybı proje sürecini etkileyeceğinde zaman etüdü yapılarak kayıp minimuma indirilmiştir. Şelale modeli uygulanırken 18 çalışan varken, SCRUM modeliyle çalışan sayısı 13 kişiye düşürülmüştür.

Firma içerisinde Çevik yöntemlerden yalın felsefeye en yakını olan SCRUM modelinin hızla yaygınlaşabilmesi için, yapılacak farklı iş birimlerinden yeni pilot projeler seçilmesi önem taşımaktadır. Bu sayede önemli başarılar yakalandıkça, ilk yapılan pilot projenin şans eseri başarı getirmediğinin ispatı olacak ve kullanıcı memnuniyeti daha da artacaktır. Çevik yöntemlerin sadece bilgi işlem kapsamında veya yazılım projelerinde değil tüm şirket genelinde yönetimsel alt yapı olarak hedeflenmesi gerekmektedir.

Firmalar, eğitime daha fazla yatırım yapmalı, çalışanların deneyimlerini paylaşacakları, iletişimin aktif olacağı ortamlar oluşturmalıdırlar. Yazılım

geliştirmede sürenin en çok kodlamaya harcanması, analiz sürecine önem verilmemesi proje sürecinde yanlış bir yolun izlendiğini göstermektedir. Gereksinim analizi hazırlama aşamasına daha çok önem verilmesi ileriki safhalarda karşılaşılan hataları azaltacak, bu durum maliyet ve zaman tasarrufu sağlayacaktır.

5S ve Metot Etüdü teknikleri sadece yazılım geliştirme ekibinde değil tüm firmada benimsenmelidir. Burada yöneticilere büyük rol düşmektedir. İlk önce yöneticiler eğitilip çalışanları yönlendirerek disiplin oluşturmaları gerekmektedir. Bu tekniklerle ilgili eğitimler verilmelidir. Tekniklerle alakalı sloganlar, afişler ve bültenler düzenlenebilir, çalışanların dikkatleri çekilebilir. İyileştirme boyutunda öneri/fikir üretme yarışmaları yapılabilir, tekniklerle ilgili günler düzenlenerek (5S günü gibi) çalışanların tamamının katılımı sağlanabilir veya tekniği en iyi uygulayan departman yarışması yapılarak çalışanlar heveslendirilebilir.

OBEYA tekniği de aynı şekilde firmanın çoğu departmanında benimsenebilir. OBEYA odası oluşturup yapılacak işler duvarlarda belli bölgelere ayırarak planlar yapılabilir. Günlük toplantılar yapılarak iş planı, gecikme olup olmadığı, neler yapıldığı hakkında bilgi aktarılarak işler takip edilebilir. Bu teknik uygulanarak zaman kaybı önlenir, çalışanların temiz ve organize bir ortamda çalışmalarıyla moral ve motivasyonu yükselir ve düzen sayesinde kalite oranı artar.

KAYNAKLAR

Abe, S., Mizuno, O., Kikuno, T., Kikuchi, N., Hirayama, M., (2006), "Estimation of project success using Bayesian classifier, Proceeding of the 28th International Conference on Software Engineering", ICSE'06, Shanghai China, 600-603.

Acar, Ö., (2008), "Extreme Programming Nedir?", 08.02.2017, http://www.kurumsaljava.com/2008/11/21/extreme-programming-nedir/

Apilioğulları, L. (2012), "Toyota Management System", 10.02.2017, http://www.lutfiapiliogullari.com/toyota-management-system/

Ataman, G., (2009), "İşletme Yönetimi", 2. Baskı, İstanbul: Türkmen Kitapevi, 109.

Aydın, M., (2007), "Eğitim Yönetimi". Ankara: Hatiboğlu Yayınları, 107.

Aydın, O., (2007), "Süreç İyileştirmede Bilgi Yönetimi Uygulamalarının Kullanılması Üzerine Bir Vaka Analizi", Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi.

Azizia, A., (2015), "Designing a Future Value Stream Mapping to Reduce Lead Time using SMED-A Case Study", Procedia Manufacturing 2, 153-158.

Baggaley, B. ve Maskell, B., (2003), "Value Stream Management For Lean Companies, Part I", Cost Management 17(2), 23-27.

Başar, A., Özkaya, A. ve Kesgin, F., (2014), "Yazılım Geliştirme Süreçlerinde Şelale Yönteminden Çevik Yaklaşıma Geçiş: Bir Teknoloji Şirketinde Uygulama", Ziraat Teknoloji A.Ş., İstanbul.

Bawden, R. ve Skerritt O., (2002), "The Concept of Process Management, The Learning Organization", 9, 3, 132.

Birgün, S. Gülen, K.,G. ve Özkan, K., (2006), "Yalın Üretime Geçiş Sürecinde Değer Akışı Haritalama Tekniğinin Kullanılması: İmalat Sektöründe Bir Uygulama", İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi, 5, 9, Bahar 2006/1, 47-59.

Birgün, S., (2006), "Aksiyomlarla Tasarım Yoluyla Deger Akısı Haritalandırma", Yöneylem Arastırması ve Endüstri Mühendisliği XXVI. Ulusal Kongresi Bildiriler Kitabı, 35-40, Kocaeli Üniversitesi, 3-5 Temmuz 2006.

Birgün, S., Gülen, K. G. and Özkan, K., (2006), "A Case Study on Eliminating Waste from the Business Processes", Proceedings of the 15th Annual World Business Congress, International Management Development Association, Sarajevo, Bosnia and Herzegovina, 41-46.

Boehm, B. W. ve Turner, R., (2003), "Balancing Agility and Discipline: A Guide for the Perplexed", Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 35-54.

Bozkurt, R., (2003), "Süreç İyileştirme", Ankara: MPM Yayınları, 12.

Byrne, A., (2015), "Yalın Dönüşüm", Melis İnan (çev.), İstanbul: Optimist Yayıncılık, 109.

Chitturi, R.M., Glew, D.J. ve Paulls, A., (2007), "Value Stream Mapping in a Jobshop", IET International Conference on Agile Manufacturing, 9-11 Temmuz, Durham, United Kingdom.

Çandur, C., (2010), "Bankacılık IT Proje Yönetimi ve Ms Project'le Proje Planlama", Yüksek Lisans Tezi, Haliç Üniversitesi.

Çetin, E., Durdu, P., (2015), "Türkiye'de Çevik Yazılım Geliştirme Üzerine Bir İnceleme", Yüksek Lisans Tezi, Kocaeli Üniversitesi.

Çırak, S., (2013), "Proje Yönetiminde Yalın ve Kısıtlar Teorisi İle Bir Uygulama", Yüksek Lisans Tezi, Marmara Üniversitesi.

Doğan, Y., Özkütük, A. ve Doğan, Ö., (2014), "Laboratuvar Güvenliğinde "5S" Yönteminin Uygulaması ve Çalışan Memnuniyeti Üzerine Etkisi", Mikrobiyol Bul Dergisi, İzmir, 48(2): 300-310.

Doğruer, M., (2014), "İş Etüdü", İstanbul: Açılım Kitap Yayınevi, 62.

Dowdle, P., Stevens, J., McCarthy, B. ve Daly, D., (2003), "Process-Based Management: The Road To Excellence", Cost Management, 17/4, 13.

Dinesh, S. ve Gupta, V., (2005), "Application of Value Stream Mapping for Lean Operations and Cycle Time Reduction: An Indian Case Study", Production Planning and Control, 16, 1.

Eren, Z., (2010), "Kamu kurumlarında süreç yönetimi: İTÜ Fen Bilimleri Enstitüsü uygulaması", Yayımlanmamış Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi.

Eyüboğlu, F., (2005), "Süreçlerle İlgili Bazı Kavram ve Yaklaşımlar Hakkında Bilgi", www.filizeyuboglu.com/yazi2

Eyüboğlu, F., (2012), "Süreç Yönetimi ve Süreç İyileştirme", İstanbul: Sistem Yayıncılık.

Gül, Z., (2006), "Yazılım Geliştirme Sürecinin İyileştirilmesi ve Türkiye Uygulamaları", Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi.

Harleen, K.,F., ve Swati, C., (2014), "A Systematic Study on Agile Software Development Methodologies and Practices", International Journal of Computer Science and Information Technologies (IJCSIT), 5, 3.

Hastie, S., (2015), Standish Group Chaos Report. www.infoq.com/articles, 2017.

Hines, P. ve Taylor, D., (2000), "Going Lean - A Guide to Implementation", Cardiff Business School, Lean Enterprise Research Centre, United Kingdom, 10.

ISO/IEC, (2006), "Information Technology, Software Process Assessment", 15504-5

Jackson, T., (2009), "5S for Healthcare (Lean Tools for Healthcare Series)", New York: Productivity Press, Taylor ve Francis Group.

Karadağ, L., (2002). "Proje yönetimi, BT proje yönetimi ve başarısızlık nedenleri", Bilgi Yönetimi, İstanbul.

Koçel, T., (2001), "İşletme Yöneticiliği", 8. Baskı, İstanbul: Beta Basım, 141.

Kürkçü, H., (2005), "Süreç ve Sistem Yaklaşımına İnsan Kaynakları Yönetim Penceresinden Bir Bakış", İstanbul: Mess Yayınları, 39.

Liker, K.J., (2005), "Toyota Tarzı 14 Yönetim İlkesi". Ümit Şensoy (çev.), İstanbul: Orhan Holding Yayınları.

Lin, W. ve Quingmin, Y., (2009), "Lean Accounting Based on Lean Production", Management and Service Science, MASS'09 International Conference, 20-22.

Meriç, A., (2011), "Yalın Üretim ile Kurumsal Kaynak Planlamasının Bütünleştirilmesi", Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi.

McDonald, T., Aken V. ve Rentes, A., (2002), "Utilising Simulation to Enhance Value Stream Mapping: A Manufacturing Case Application", International Journal of Logistic: Research and Applications, 5, 2.

Mendel, D., (2009), "BPM best practice: Enterprise business process architecture", https://community.emc.com/docs/DOC-3639, 2017.

Nalbant, S., (2012), "Yazılım Geliştirme Sürecinin Verimliliğini Arttırmak: Bir Bilgi Sistemi Önerisi", Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi.

Neumann, S., Probst, C. ve Wernsmann, C., (2003), "Continuous process management. Process Management: A Guide for the Design of Business", 233-251.

Newton, R., (2015), "Adım Adım Proje Yönetimi", İlker Gülfidan(çev.), İstanbul: Optimist Yayıncılık, 17.

Nizam, A., (2015), "Yazılım Proje Yönetimi", 2. Basım, İstanbul: Papatya Yayıncılık, 33.

Nomak, A. ve Durmuşoğlu, B., (2003), "Bir Hücresel Üretim Ortamında, Üretim Planlama ve Kontrol Sistemlerinin Benzetim Analizi", İTÜ Mühendislik Dergisi, 2, 5.

Palmer, S.R. ve Felsing, M., (2001), "A Practical Guide to Feature-Driven Development", 1.Basım, Pearson Education, USA, 57-63.

Prabhu, B. V., Surekha, A., Holla, A.J. ve Patel, K. M., (2008), "Value Stream Mapping of Truck Operations: A Case Study", South Asian Journal of Management, April-June, 107-115.

Rhinesmith, S., (2000), "Yöneticinin Küreselleşme Rehberi", Gülden Şen (çev.), Sabah Kitapları, İstanbul, 110, 20-21.

Rother, M. ve Shook, J., (1999), "Learning to See", Versiyon 1.2., The Lean Enterprise Institute Inc, Brookline, Massachusetts, USA.

Seth, D. ve Gupta, V., (2005), "Application of Value Stream Mapping for Lean Operations and Cycle Time Reduction: An Indian Case Study", Production Planning & Control, 16/1, 44-59.

Seth, D. ve Goel, D., (2007), "Application of Value Stream Mapping (VSM) for Minimization of Wastes in the Processing Side of Supply Chain of Cottonseed Oil Industry in Indian Context", Journal of Manufacturing Technology Management, 19(4): 529-550.

Simons, D., Zokaei, K., (2005), "Application of Lean Paradigm in Red Meat Processing", British Food Journal, United Kingdom, 107, 4.

Sönmez, Z., (2013), "Altı Sigma Metodolojisi ile Süreç İyileştirme ve Hizmet Sektöründe Bir Uygulama", Yüksek Lisans Tezi, İstanbul Kültür Üniversitesi.

Sutherland, J. ve Schwaber, K., (2016), "SCRUM Guides" www.scrumguides.org/docs/, 2017.

Svensson, R., (2006), "Successful Software Projects and Products". Blekinge Institute of Technology Master Thesis Software Engineering.

Şahin, H., (2002), "Proses Tabanlı Kalite Yönetim Sistemi", Ankara: Polimer Matbaacılık, 6.

Şeker, S.E., (2015), "Yazılım Geliştirme Modelleri ve Sistem/Yazılım Yaşam Döngüsü", YBS Ansiklopedi Dergisi, İstanbul, 2, 3.

Tapping, D., Luyster, T. ve Shuker, T., (2002), "Value Stream Management Eight Steps to Planning, Mapping and Sustainining Lean Improvements". New York: Productivity Inc., USA.

Tunç, M., (2016), "Kamu Kurumlarında Yalın Yönetim: Sosyal Güvenlik Kurumu (SGK) Örneği", Yüksek Lisans Tezi, Beykent Üniversitesi.

Özcan, E., (2013), "Ofiste Temel Ergonomik İyileştirmeler için", 11.02.2017, http://www.ofisteyasam.com/2013/08/12/ofiste-temel-ergonomik-iyilestirmeler-icin/

Özveri, O. ve Güçlü, P., (2015), "Değer Akış Haritalamada Analitik Hiyerarşi Süreci (AHP) Uygulanması", Uluslararası Alanya İşletme Fakültesi Dergisi, Antalya, 7/1, 1-12.

Womack, J.P. ve Jones, D.T., (2010), "Yalın Düşünce", Oygur Yamak (çev.), İstanbul: Optimist Yayınları, 23-433.

Yaman, Ö., (2007), "Örgütlerde Yalın Yönetim", Yüksek Lisans Tezi, İnönü Üniversitesi.

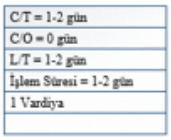
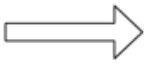
Yitmen, M., (2015), "SCRUM Bir Dönüşüm Hikayesi Agile Proje Yönetimi", İstanbul: Seçkin Yayıncılık, 21.

Yurdugül, U., (2010), "Değer Akışı Haritalama Yöntemi ve Bir Uygulama", Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi.

Zoroğlu, B., (2013), "Yalın Üretim", 1.Basım, Ankara: Sage Yayıncılık, 72-80.

EKLER

Ek-A Değer Akış Haritalamada Kullanılan Semboller

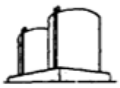
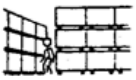
Malzeme Sembolleri	Tanımı	Açıklama
	Üretim Prosesi	Bir Proses kutusu bir akış alanına karşılık gelir. Bütün prosesler belirtilmelidir. Ayrıca üretim kontrol gibi departmanları göstermek için kullanılır.
	Dış Kaynaklar	Müşterileri, tedarikçileri ve dışarda gerçekleştirilen üretim proseslerini göstermek için kullanılır.
	Bilgi Kutusu	Üretim prosesi,departman, müşteri vb. ilgili bilgileri kaydetmek için kullanılır.
	Stok	Miktar ve zaman not edilmelidir.
	Kamyonla Sevkiyat	Sevkiyat sıklığı not edilmelidir.
	Üretilen Malzemenin İtme ile Hareketi	Bir sonraki prosesin ihtiyacından önce üretilen ve ileriye doğru itilen malzeme; genellikle çizelgeye bağlıdır.
	Bitmiş Ürünün Müşteriye Hareketi	

	Süpermarket	Önceki prosesin üretimini çizelgelemek için kullanılan kontrollü parça stoğudur.
	Çekiş	Malzeme çekişleri genellikle süpermarkettendir.
	Prosesler arasında FIFO prensibine göre, kontrollü mikttarda malzeme transferi	Prosesler arasında miktarı sınırlayan ve FIFO akışını sağlayan bir araçtır.

Bigi Sembolleri	**Tanımı**	**Açıklama**
	Manuel Bilgi Akışı	Üretim çizelgesi veya sevkiyat çizelgesi
	Elektronik Bilgi Akışı	Örneğin EDI (Elektronik Veri Değişimi) ile
	Bilgi	Bilgi akışını tanımlar.
	Üretim Kanbanı	Prosese hangi parçadan kaç tane üreteceğini söyleyen ve bunu yapması için izin veren bir kart ya da araçtır.
	Çekme Kanbanı	Malzeme taşıyıcıya parça transfer etmesi emrini veren kart ya da araçtır.
	Sinyal Kanbanı	Yeniden sipariş noktasına ulaşıldığında sinyal verir ve yeni bir parti üretilir.
	Sıralı-Çekme Topu	Önceden belirlenmiş tip ve miktarda hemen üretim yapması emrini verir. Süpermarket kullanmadan altmontaj prosesleri için bir çekme sistemidir.
	Kanban Kutusu	Kanbanların toplandığı ve dağıtım için tutulduğu yer.
	Yük Seviyelendirme	Bir zaman dilimi içinde üretim hacmini ve ürün karmasını seviyelendiren bir araçtır.

Kaynak: Rother ve Shook, (1999)

Ek-B Metot Etüdünde Kullanılan Semboller ve Açıklamaları

İşlem  Yukarıdaki işaret yandaki gibi örnek işlemin türünü belirtir.	 Çivi çakma	 Karıştırma	 Form oluşturma
Taşıma  Bu ok yanda örneklerdeki gibi taşıma işlerini içerir	 El ile itilen araç ile malzeme taşıma	 Konveyörle malzeme taşıma	 El ile yürüyerek, parça malzeme, dosya vb. taşıma
Depolama  Üçgen işareti yandaki örneklerdeki gibi depolamayı gösterir	 Ham malzemenin stok alanındaki stoğu	 Bitmiş parçaların paletler üzerinde stoklanması	 Dökümanların koruma amaçlı saklanması

Gecikme  Büyük D harfi ile örneklerdeki gecikme nedenleri belirtilir.	 Asansör bekleme	 Bir kutu içerisinde tesviye-el işi için bekleyen parça grubu	 İşlenmeyi, yazılmayı bekleyen kağıtlar
İnceleme  Kare işareti yandaki örneklerdeki gibi muayene-yoklama faaliyetlerini içerir.	 Nicelik ve nitelik olarak malzemenin kontrolü	 Kazandaki buhar sıcaklığının ve basıncının ölçülmesi	 Basılı kağıt, formların incelenmesi, gözden geçirilmesi

ÖZGEÇMİŞ

3 Temmuz 1990 tarihi, İstanbul ili Bakırköy ilçesi doğumluyum. İlköğretimi Bakırköy ilçesinde Pilot Cengiz Topel İ.Ö.O'nda, Liseyi Çapa ilçesinde Şehremini Süper Lisesi'nde tamamladım. Lisansta İstanbul Kültür Üniversitesi İşletme bölümünden 2013 yılında ve Matematik – Bilgisayar (Çift Anadal) bölümünden 2014 yılında mezun oldum. Özel bir sigorta şirketinde Bilgi Teknolojileri departmanında 'Uzman Sistem Analisti' olarak çalışmaktayım. 2014 yılında Beykent Üniversitesi, Fen Bilimleri Enstitüsü, Endüstri Mühendisliği Yüksek Lisans eğitimine başladım.

Özel ilgi alanlarım, proje yönetimi, süreç iyileştirme teknikleri, çevik yazılım geliştirme modelleri ve karar verme süreçleridir.

Burcu TANRIKUT ÇERKEZOĞLU