KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

YÜKSEK LİSANS TEZİ

YAZILIM GELİŞTİRME ORGANİZASYONLARI İÇİN HARMANLANMIŞ SCRUM MODELİ

ESRA ÇETİN

KOCAELİ 2016

KOCAELİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ

ANABİLİM DALI

YÜKSEK LİSANS TEZİ

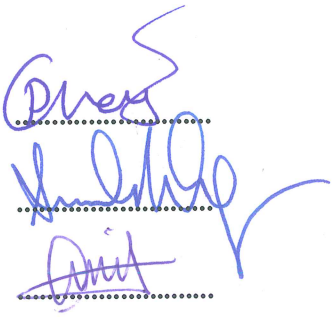
YAZILIM GELİŞTİRME ORGANİZASYONLARI İÇİN HARMANLANMIŞ SCRUM MODELİ

ESRA ÇETİN

Yrd.Doç.Dr. Pınar ONAY DURDU
Danışman, Kocaeli Üniversitesi

Prof.Dr. Semih BİLGEN
Jüri Üyesi, Yeditepe Üniversitesi

Yrd.Doç.Dr. Davut İNCEBACAK
Jüri Üyesi, Kocaeli Üniversitesi



Tezin Savunulduğu Tarih: 16.06.2016

ÖNSÖZ VE TEŞEKKÜR

Bu tez çalışmasında, giderek gelişen yazılım dünyasında kullanılan yazılım geliştirme yöntemlerinden biri olan Scrum yazılım geliştirme yöntemini uygulayan organizasyonlar incelenerek, yazılım profesyonellerinin sorunları, memnuniyetleri ve yöntemi uygulama şekilleri analiz edilmeye çalışılmıştır. Çalışma kapsamında başlangıçta gerçekleştirilen alan yazın araştırması ve anket çalışması ile geleneksel yazılım geliştirme yöntemlerine alternatif olarak geliştirilen çevik yazılım geliştirme yöntemlerinden en yaygın olarak kullanımı görülenin Scrum metodolojisi olduğu belirlenmiştir. Daha sonrasında organizasyonlarında Scrum yazılım geliştirme yöntemini uygulayan profesyonellerin görüşleri alınarak, Scrum sürecine, pratiklerine, kavramlarına ve araçlarına bakış açıları gözlemlenmiştir. Alan yazında ve incelemeler sırasında organizasyonların Scrum metodolojisini kendi bünyelerinde gelişen ihtiyaçları ve tecrübeleri doğrultusunda şekillendirmeye çalıştığı, modele yeni özellikler ekledikleri veya modeldeki bazı özellikleri uygulamadıkları tespit edilmiştir. Alan yazında Scrum ama (Scrum but) olarak nitelendirilen bu durum, bazı araştırmacılar tarafından tehlikeli bir durum olarak belirtilmesine karşın, bazıları da organizasyonların Scrum metodolojisini olduğu gibi uygulamaları yerine kendilerine uygun olarak uyarlamalarının daha verimli olacağını savunmaktadırlar. Bu noktadan hareket ile Scrum yazılım geliştirme yöntemini uygulayan organizasyonlarda kullanılabilecek bir yazılım geliştirme modeli olarak Harmanlanmış Scrum Modeli önerilmektedir.

Tez çalışmam boyunca, yol göstericim olan, konum ile ilgili fikir ve tecrübelerini benden esirgemeyen, bu zorlu süreçte beni sürekli destekleyen, danışmanım sayın Yrd.Doç.Dr. Pınar ONAY DURDU'ya başından sonuna her adımdaki tüm yardımları için gönülden teşekkür ediyorum.

Çalışma süresince varlığı ile beni yüreklendiren eşime, zor dönemlerimde yanımda olan aileme de ayrıca teşekkür ediyorum. Son olarak da anket ve değerlendirmeleri titizlikle tamamlayarak, çalışmanın güvenilirliği konusunda bana destek sağlayan tüm Scrum takım üyeleri ve organizasyonlarına minnetlerimi sunarım.

Temmuz – 2016 Esra ÇETİN

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

Kısaltmalar

BT	: Bilgi Teknolojileri
DSDM	: Dynamic System Development Methodology (Dinamik Sistem Geliştirme Modeli)
FDD	: Future driven development (Özellik Güdümlü Geliştirme)
KN	: Kategori no
MSF	: Microsoft Solution Framework (Microsoft Çözüm Çerçevesi)
RUP	: Rational Birleşik Süreç
O1	: Organizasyon 1
O1_SM	: Organizasyon 1 Scrum Master
O1_TU	: Organizasyon 1 Takım Üyesi
O2	: Organizasyon 2
O2_SM	: Organizasyon 2 Scrum Master
SM	: Scrum Master
TU	: Takım Üyesi
XP	: Extreme Programming (Uç Programlama)

YAZILIM GELİŞTİRME ORGANİZASYONLARI İÇİN HARMANLANMIŞ SCRUM MODELİ

ÖZET

Günümüz iş dünyasının rekabetçi koşulları özellikle yazılım endüstrisindeki organizasyonların geliştirdikleri ürünleri hızla pazara sunmalarını, aynı zamanda değişen gereksinimlere karşılık vermelerini gerektirmektedir. Geleneksel yazılım geliştirme modellerinin yerini almaya başlayan çevik yazılım geliştirme modelleri düşük maliyet, yüksek üretkenlik ve kalite vaatleri göz önünde bulundurularak pek çok yazılım organizasyonu tarafından son yıllarda uygulanmaya başlanmıştır. Bu çalışma kapsamında, çevik yaklaşımların organizasyonlarda kullanımı ve vaat edilen avantajları ne kadar sağladığını belirlemeye yönelik olarak bir anket çalışması hazırlanmış ve bu çalışmanın sonucunda en yaygın kullanılan çevik yöntemin Scrum metodolojisi olduğu belirlenmiştir. Sonrasında organizasyonlarında Scrum kullanan profesyoneller ile görüşmeler sağlanarak alınan geribildirimler doğrultusunda organizasyonların Scrum yazılım geliştirme modelini kendi gereksinimlerine göre şekillendirdikleri ve İdeal Scrum modelinden farklı özelliklere sahip olarak uyguladıkları tespit edilmiştir. Alan yazında Scrum ama (Scrum but) olarak nitelendirilen bu durumun Scrum metodolojisinin ana felsefesinden uzaklaşmayacak şekilde kontrollü olarak organizasyonların yararına olacak şekilde uygulanmasını sağlama amaçlı olarak Harmanlanmış Scrum Modeli önerilmiştir. İdeal Scrum Modeli ile bir çok ortak noktası olan Harmanlanmış Scrum modeli, farklı pratikleri de içinde barındıran bir model olarak önerilmektedir. Tez çalışmasının ana fikri olan Harmanlanmış Scrum modelinin tasarımı ve modellenmesi sonrasında, bir değerlendirme hazırlanarak profesyonellere sunulmuş ve görüşleri doğrultusunda model üzerinde iyileştirmeler yapılmıştır. Harmanlanmış modelde yer alması önerilen proje lideri rolü, hafta arası Scrum toplantısı ve Sprint gözden geçirme süresi kavramları profesyoneller tarafından benimsenerek, modelin Scrum kullanan organizasyonlarda uygulanabilir olduğu ile ilgili olumlu geri bildirim sağlanmıştır.

Anahtar Kelimeler: Çevik Yazılım Geliştirme, Scrum Metodolojisi, Scrum Pratikleri, Yazılım Geliştirme, Yazılım Süreç Modelleri.

BLENDED SCRUM MODEL FOR SOFTWARE DEVELOPMENT ORGANIZATIONS

ABSTRACT

Software organizations have to develop software in short time and respond changing requirements due to the competitiveness of the market. In recent years, organizations have started to implement agile software development models which promise reduced cost, increased productivity and quality. In the scope of this research, a survey was conducted to determine the use of agile methods among software organizations and to investigate whether they can achieve the promised advantages. Based on the results of the survey, Scrum was determined as the most commonly used agile development approach. Afterwards, two organizations which were using Scrum as a software development methodology were investigated as a case study. Interviews were conducted with the professionals in these organizations and it was observed that they adapt and modify Scrum according to their own needs which is called as Scrum-but in the literature rather than directly adopting it. Blended Scrum model is proposed in this study to overcome uncontrolled Scrum-but issues. Blended Scrum model has several common practices with the original Scrum and it has some extensions such as project leader role, mid-week Scrum meetings and additional Sprint review time. In addition, the practices of proposed Blended Scrum Model were evaluated with Scrum practitioners and gathered positive feedback.

Keywords: Agile Software Development, Scrum Methodology, Scrum Practices, Software Development, Software Process Models.

GİRİŞ

Günümüz iş dünyasının rekabetçi koşulları özellikle yazılım endüstrisindeki organizasyonların geliştirdikleri ürünleri hızla pazara sunmalarını, aynı zamanda değişen gereksinimlere karşılık vermelerini gerektirmektedir. Yazılım projelerinde ürünler planlanan bütçe ve zaman kriterlerine göre müşteri ihtiyaçlarını belirlenen şekilde karşılayabilme amacıyla geliştirilmektedir (Bernal ve diğ., 2013; Başar ve diğ., 2015). Bu amaçla geliştirilen bir çok farklı yazılım geliştirme süreci mevcuttur. Bu süreçler yazılım dünyasında geleneksel yazılım geliştirme ve çevik yazılım geliştirme süreçleri olarak iki ana başlık altında belirtilmektedir. Gelişen dünyada üretilen yazılımların mümkün olduğunca kısa süre içerisinde teslim edilmesinin ve değişen gereksinimlere kolay uyum sağlamasının beklenmesi sebebi ile bir çok organizasyon geleneksel yazılım geliştirme süreçleri yerine çevik yazılım geliştirme süreçlerini kullanmaya başlamıştır (Agile Manifesto, 2001; Başar ve diğ., 2015). Türkiye'de ve dünyanın bir çok bölgesinde en çok uygulanan çevik yazılım yöntemlerinden biri olarak karşımıza çıkan Scrum yazılım geliştirme yönteminin (Agile Türkiye, 2013) organizasyonlar tarafından nasıl uygulandığına yönelik farklı ülkelerde gerçekleştirilmiş çalışmalar vardır (Abrahamsson ve Salo, 2008; Pries-Heje, J. ve Pries-Heje, L., 2011).

Bu tez çalışması kapsamında Scrum yazılım geliştirme yöntemini kullanan organizasyonlarda pratik olarak uygulanabilecek özellikleri içerecek şekilde hazırlanan Harmanlanmış Scrum modeli önerilmektedir. Çevik yazılım geliştirmenin öne çıkan modeli Scrum yazılım geliştirme yöntemi ile ilgili daha önce biri büyük ve biri orta ölçekli olmak üzere 2 farklı organizasyondan yazılım profesyonelleri ile gerçekleştirilen görüşmelerden elde edilen bulgular ve konu ile ilgili yapılan anket çalışmasının verileri göz önünde bulundurulduğunda, teoride belirlenen Scrum yazılım geliştirme modeli ile pratikte organizasyonlarda uygulanan Scrum yazılım geliştirme yönteminin karşılaştırması yapılmaktadır. Görüşme ve anket sonuçlarına göre, bir çok organizasyon içerisinde Scrum yazılım geliştirme modeli,

organizasyonların büyüklüğüne, ihtiyaçlarına ve yöneticilerine göre değişiklik göstererek diğer bir deyişle organizasyona uyarlanarak, farklı şekillerde uygulanmaktadır. Alan yazında da bu durum araştırmacılar tarafından Scrum ama (Scrum but) olarak nitelendirilmektedir (Schwaber, 2011; Sutherland, 2009). Scrum modelinin uyarlanmadan orijinal hali ile uygulanması gerektiğini savunanlar olduğu gibi (URL-2, 2016; Jeffries, 2013) organizasyonların yapısına göre uyarlanmasının ya da iyi pratikler ile entegrasyonunun uygun olduğunu savunan çalışmalar da yer almaktadır (Cho, 2009; Hayata ve Han, 2011). Bu çalışmanın amacı, Scrum yazılım geliştirme yönteminin, organizasyonlarda nasıl uygulandığını tespit ederek, teorideki Scrum metodolojisi ile organizasyonların uyguladıkları Scrum modelinin farklarını gösterebilmektir. Çalışma kapsamında farklı organizasyonlarda uygulanan farklı özellikleri ile henüz uygulanamayan ancak profesyoneller tarafından uygulanırsa daha iyi olabileceğine dair görüş bildirilmiş özellikleri tek bir çatı altında toplayarak, organizasyonlarda uygulanabilecek Harmanlanmış Scrum modelini ortaya çıkarmak amaçlanmaktadır. Görüşmeler sonucunda organizasyonların Scrum metodolojisini uygularken edindikleri tecrübeler göz önüne alınarak, profesyonellerin Scrum modeli üzerindeki algıları belirlenmiştir.

1990'lı yılların başlarında Ken Schwaber tarafından geliştirilen model (Schwaber, 1995) İdeal Scrum Modeli olarak tanımlandığında, çalışma kapsamında anket ve görüşme sonuçlarına dayanarak, görüşülen profesyonellerden alınan geri bildirimler doğrultusunda oluşturulan Harmanlanmış Scrum modeli ise, farklı organizasyonların pratikte uyguladıkları Scrum uygulamaları ile İdeal Scrum modelinin özelliklerini birlikte içeren bir model olarak öngörülmektedir. Harmanlanmış Scrum modeli ile, İdeal Scrum modelinde yer alan özelliklerin yanı sıra, İdeal modelde bulunmadığı halde organizasyonlarında Scrum yazılım geliştirme yöntemini uygulayan profesyonellerin tecrübeleri ile uygulamaya başladıkları bazı özellikler tek bir model içerisinde birleştirilmektedir. Görüşme ve anketlerden elde edilenler dışında, Scrum yazılım geliştirme sürecini iyileştirmek adına, Harmanlanmış Scrum modeli üzerinde tanımlanan farklılıklar genel süreç yapısını olumsuz olarak etkilemeden eklenmeye çalışılmıştır. İdeal Scrum modelinde bulunan Scrum pratiklerinin ve kavramlarının bir kısmı Harmanlanmış Scrum modelinde de korunurken bazıları ise kullanılmamaktadır.

Bu tez çalışması sekiz bölümden oluşmaktadır. Giriş bölümünde tezin konusu ve amacı açıklanarak, tezin bölüm içeriklerine ait konular özetlenmektedir. Birinci bölümde yazılım mühendisliği, yazılım geliştirme yöntemleri, geleneksel yazılım geliştirme ve çevik yazılım geliştirme yöntemleri ile ilgili alan yazın çalışmasına yer verilmektedir. İkinci bölümde çevik yazılım geliştirme yöntemleri içerisinde yer alan Scrum yazılım geliştirme modelinin metodolojisi, fazları, süreci, rolleri, pratikleri ve Scrum ile ilgili yaşanan çeşitli problemler anlatılmaktadır. Üçüncü bölümde tez çalışması kapsamında kullanılan araştırma stratejisi ve tasarımı, veri toplama teknikleri, veri analizi ve katılımcılar ile organizasyonlarına ait bilgiler, araştırma yöntemi başlığı altında detaylı olarak açıklanmaktadır. Dördüncü bölümde ise anket ve görüşmeler ile edinilen bulgular analiz edilmektedir. Beşinci bölümde önerilen Harmanlanmış Scrum modeli ise, tüm süreci ve fazları ile sunulmaktadır. Altıncı bölümde Harmanlanmış Scrum modeli uygulanabilirliği bir değerlendirme aracılığı ile tespit edilmekte, değerlendirme sonuçları doğrultusunda model üzerinde yapılan güncelleme ve iyileştirmeler ise yedinci bölümde yer almaktadır. Son bölümde yer alan sonuç bilgisi kısmında, ise tez çalışmasının genel sonuçları ile ilgili bilgiler ve gelecekte yapılabilecek çalışmalara ait detaylar sunulmaktadır.

1. ALANYAZIN ARAŞTIRMASI

Bu bölüm, bu tez çalışmasının gerçekleştirilebilmesi için gerekli alan yazın araştırmasını sunmaktadır. Yazılım geliştirme, çevik yazılım geliştirme ve geleneksel yazılım geliştirme kavramları ile ilgili temel bilgiler açıklanmaktadır.

1.1. Yazılım Mühendisliği ve Yazılım Geliştirme

Bir yazılımın üretiminden kullanıcılar tarafından kullanımına kadar geçirdiği evrelerin tümü olarak tanımlanabilen Yazılım Yaşam Döngüsü (Software Life Cycle) planlama, analiz, tasarım, kodlama, test, teslimat ve bakım safhalarını içermektedir. Bu safhalar içerisinde, planlama aşamasında müşterinin gereksinimlerinin belirlenmesi, analiz aşamasında belirlenen gereksinimlerin netleştirilmesi ve belgelenmesi, tasarım aşamasında yazılımın kullanıcıya bakan yüzü olan kullanıcı arayüzünün ve yazılım özelliklerinin belirlenmesi, kodlama aşamasında yazılımın kodlanması, test aşamasında kodlanan yazılımın testi, beklenen ve gözlenen sonuçların karşılaştırılması, teslim aşamasında yazılımın müşteriye teslimi, bakım aşamasında ise kullanıcılar tarafından kullanılan yazılımın tespit edilen hatalarını düzeltmesi işlemleri yapılmaktadır (Pressman, 2005). Yazılım projelerinde hedeflenen, belirli süre içerisinde kısıtlı bir bütçe ile, müşteri tarafından kabul edilebilir kaliteli bir ürün geliştirmektir (Bernal ve diğ., 2013). Geleneksel yazılım geliştirme yöntemlerinde tüm safhalar birbiri ardına düzenli olarak sıralanır, bir safha tamamlanmadan bir sonraki safhaya geçilmez. Bu özellikleri ile yazılım sürecinin uzamakta ve son safhada görülen hataların düzeltilmesine imkan verilmemektedir (Sommerville, 2000). Yazılımın gerçek kullanıcısı olacak müşteriler, sadece planlama ve analiz safhalarında yer alır (Bernal ve diğ., 2013). Geleneksel yöntemlere alternatif olarak geliştirilen çevik yöntemlerde ise müşteriler yazılım sürecinin her aşamasına dahil edilir ve gereksinimler üzerinde yazılım sürecinin son aşamasına kadar değişiklik yapılmasına izin verilir. Birkaç haftada bir müşteriye çalışan bir yazılım sunulur ve bu ilerlemenin en önemli ölçüsü olarak görülür (Agile Manifesto, 2001).

1.2. Geleneksel Yazılım Geliştirme

Geleneksel yazılım geliştirme süreçlerinde gereksinim belirleme, analiz, tasarım, kodlama, test, teslimat ve bakım safhalarının tümü, Yazılım Yaşam Döngüsü (Software Life Cycle) içerisinde yer almaktadır. Yazılım geçirdiği bu evreler içerisinde analiz aşamasında gereksinimler belirlenmekte, tasarım aşamasında yazılım ara yüzü oluşturulmakta, daha sonra ise ürünün kodlanmasına geçilmektedir. Test aşamasında çeşitli yazılım testleri ile ürünün beklenenleri karşılama durumu kontrol edilir. Test aşaması tamamlanarak onaylanan yazılımlar müşteriye teslim edilmekte, teslimat sonrası yaşanan sorunlar ise bakım safhasında ele alınmaktadır (Pressman, 2005)

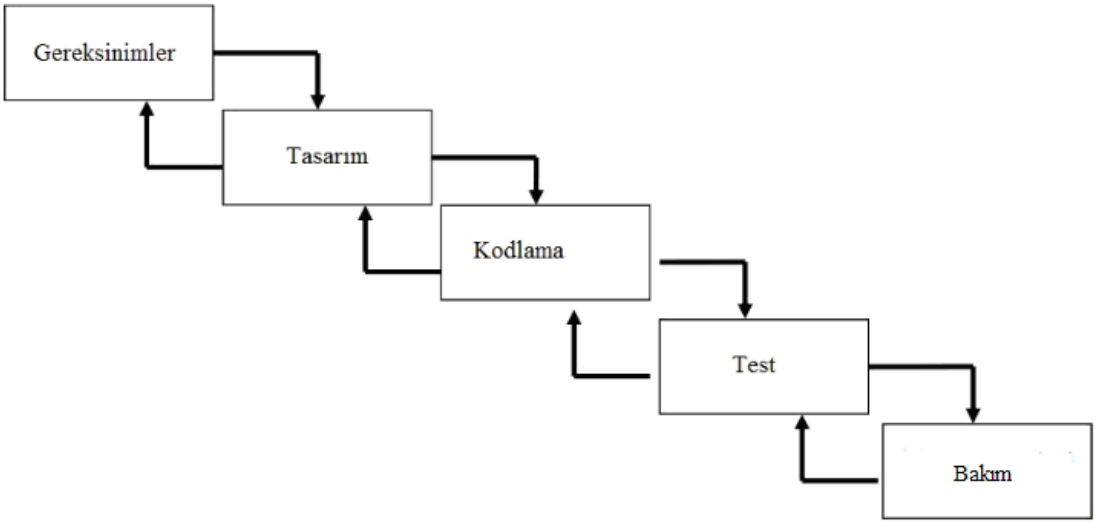
1.2.1. Geleneksel yazılım geliştirme modelleri

Geleneksel yazılım süreçlerinde tüm gereksinimler proje başlangıcında müşteriden alınmaktadır. Tüm gereksinimleri detaylı bir şekilde belirlenmiş projeler için uygun olan bu yapı, proje içerisinde gereksinimlerinde değişiklik gösterecek projeler için uygun değildir. Geleneksel yazılım geliştirme yöntemlerinden bazıları aşağıda sıralanmaktadır. Bunlar arasında en klasik yazılım geliştirme modeli şelale modelidir (Pressman, 2005; Schwaber, 2004).

- Şelale modeli
- Spiral model
- V model

1.2.1.1. Şelale modeli (Waterfall model)

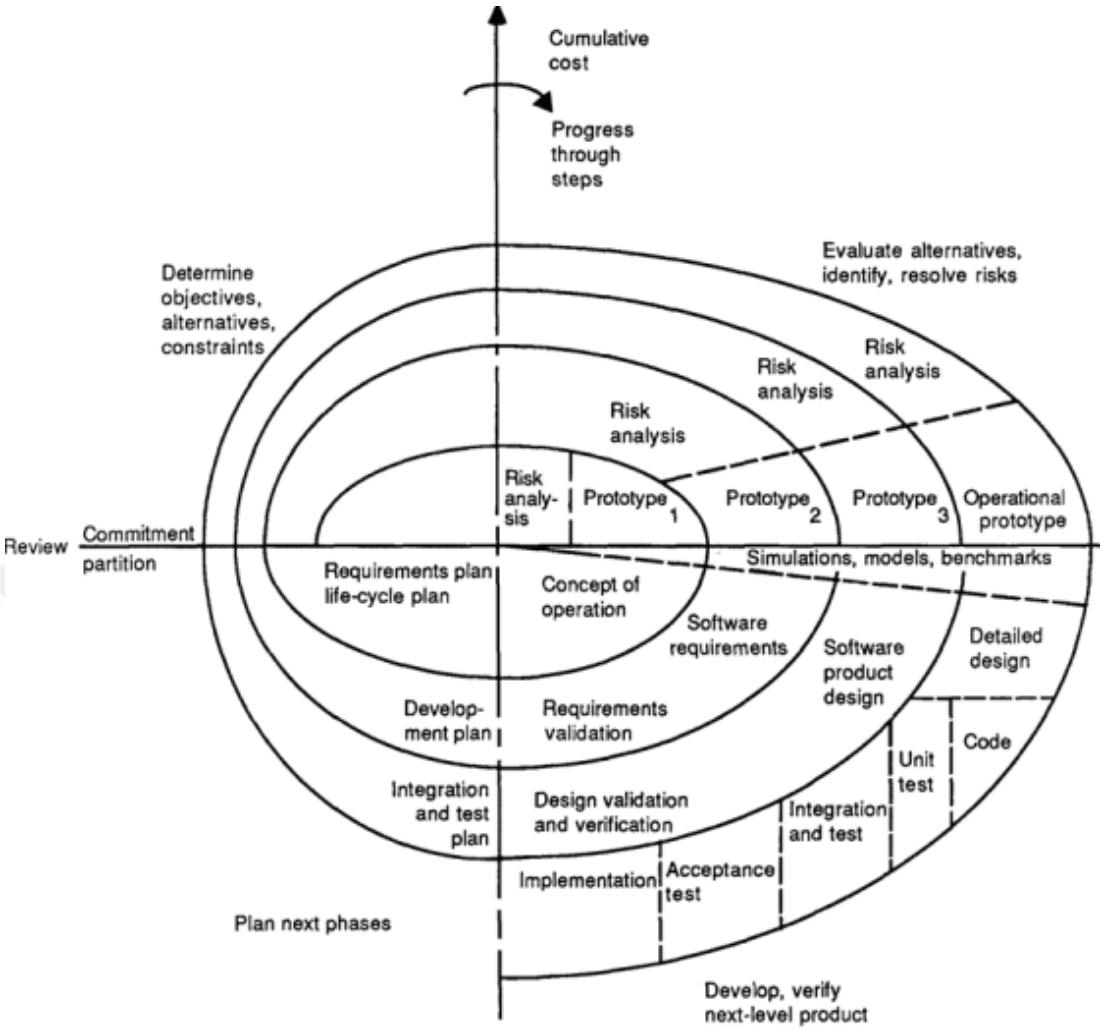
Şelale modeli beş adet işlem adımından oluşmaktadır. Bu aşamalar gereksinim analizi, tasarım, kodlama, test ve bakım aşamalarıdır. Bir adım tamamlanmadan diğer adıma geçilmemekte ve her safha sonunda doküman oluşturulma işlemi yapılmaktadır (Pressman, 2005). Test aşamasının sonlara bırakıldığı bu modelde, müşteri ihtiyaçlarına cevap vermek zorlaştırmaktadır (Sommerville, 2000). Şelale modelinde yer alan aşamalar Şekil 1.1'de izlenmektedir (Pressman, 2005).



Şekil 1.1. Şelale modeli (Royce, 1970)

1.2.1.2. Spiral (Sarmal model)

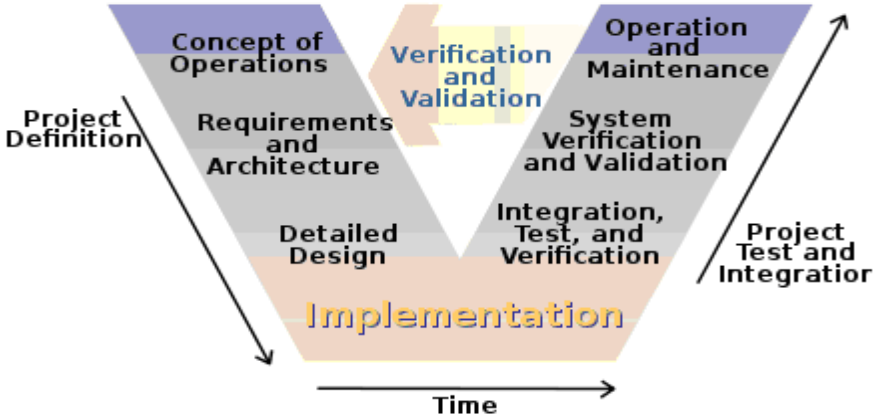
1986 yılında Barry Boehm tarafından geliştirilen Spiral (Sarmal) model temel olarak dört aşamadan oluşmaktadır (Boehm, 1988) Birinci aşamada sistem ifade edilir ve uygulama alternatifleri, objeleri ile kısıtları araştırılır. İkinci aşamada alternatifler tanımlanarak, riskler belirlenir. Geliştirme ve tanımlama işlemleri üçüncü aşamada yapılırken, diğer fazların planlaması ise dördüncü aşamada yapılır (Wasson, 2005). Bu aşamaların tamamı Şekil 1.2'de izlenmektedir.



Şekil 1.2. Spiral model (Boehm, 1988)

1.2.1.3. V model

V model içerisinde bir tarafta doğrulama evreleri ve bir tarafta geçerleme evreleri bulunmaktadır. Model içerisinde doğrulama evresinin her bir seviyesi, geçerleme evresindeki bir seviyeye karşılık gelecek şekilde yerleştirilmiştir. Doğrulama evrelerinde gereksinim analizi, sistem tasarımı, mimari tasarım ve modül tasarımı yer alırken; geçerleme evrelerinde birim test, entegrasyon testi, sistem testi ve kullanıcı kabul testi yer almaktadır. V modelde testler sırasında tespit edilen hataların düzeltilmesi için ilgili düzeydeki doğrulama seviyesine geçilmektedir. V model içerisinde yer alan doğrulama ve geçerleme evreleri Şekil 1.3'te izlenmektedir (Johansson & Bucanac, 1999; URL-5, 2016).



Şekil 1.3. V model (URL-5, 2016)

1.3. Yazılım Geliştirme Problemleri

Günümüzde yazılım projelerinin bir çoğu istenen başarıyla tamamlanamamaktadır (Labuschagne ve Marnewick, 2009; Galorath, 2012). Standish Group Bilgi Teknolojileri (BT) alanında yaşanan proje başarısızlıkları ile ilgili düzenli çalışmalar (Dominguez, 2009) yürütmektedir. Yayınladığı "Chaos" raporlarında projeler başarılı, zorlanmış (ek maliyetli) (zaman, bütçe ya da proje gereksinimlerinin tümünü karşılayamamış) ve başarısız olarak sınıflandırılmaktadır. Tablo 1.1'de görüldüğü üzere, 1994'ten günümüze dek olan raporların sonuçlarına göre, başarılı proje oranında artış yaşanırken başarısız proje oranlarında düşüş yaşanmaktadır. Tablo 1.1 dikkatlice incelendiğinde 2002'ye kadar başarısız proje oranında düşüş yaşandığı, sonrasında ise yeniden artış olduğu görülmektedir. Başarılı projelerde gözlemlenen artış ise gelişen teknoloji, sahip olunan daha fazla bilgi birikimi, daha iyi araç ve teknikler göz önünde bulundurulduğunda oldukça düşük kalmaktadır. Bunda proje karmaşıklıklarının artması ve teslimat sürelerinin kısalması gibi faktörlerin etkisi de oldukça fazladır (Dominguez, 2009). Türkiye'de de buna benzer bir durum izlenmekte ve yazılım projelerinde başarı oranı % 50'nin altında olduğu düşünülmektedir (Agile Türkiye, 2013).

Tablo 1.1. Chaos raporu verileri (Dominguez, 2009)

	1994	1996	1998	2000	2002	2004	2006	2009	2011
Başarılı	16%	27%	26%	28%	34%	29%	35%	32%	37%
Ek maliyet	53%	33%	46%	49%	51%	53%	46%	44%	42%
Başarısız	31%	40%	28%	23%	15%	18%	19%	24%	21%

Yazılım projelerinin hedeflenen kapsamda, zamanda ve bütçede tamamlanması organizasyonlar için çok önemlidir. Projelerin başarılı olması için en iyi uygulamaların veya sistem geliştirme modellerinin organizasyon içerisinde uygulanıyor olması gerekmektedir. Yazılım projelerinde olumsuzlukların yaşanmaması için yazılım geliştirme süreci günümüz rekabet koşullarına uygun şekilde yürütülmelidir. Bu amaçla geleneksel yazılım geliştirme yaşam döngüsü modellerine alternatif olarak çevik yöntemler (Agile Alliance, 2001) geliştirilmiştir. Çevik yazılım geliştirme yöntemleri hantal ve bürokratik olarak eleştirilen geleneksel, plana dayalı yazılım geliştirmeye karşılık 2001 yılında bir grup araştırmacı tarafından yazılım geliştirme için daha iyi bir yol olarak önerilmiştir. Çevik yöntemlerin yazılım kalitesini arttırdığı (Livermore, 2007) paydaşlar arası iletişim (Abrahamsson ve diğ., 2008) ve koordinasyonu (Hope ve diğ., 2012) iyileştirdiği, üretkenliği arttırdığı (Jakobsen ve diğ., 2008) ve günümüzde yazılım sistemlerini geliştirme, sürdürme ve destekleme için en iyi yol olduğu (Mary ve Suganya, 2010) iddia edilmektedir. İddia edilen bu özelliklerine rağmen çevik yazılım geliştirme yöntemleri tüm organizasyonlar tarafından henüz uygulanmamaktadır (Lundqvist ve Srinivasan, 2009).

1.4. Çevik Yazılım Geliştirme

Çevik yazılım geliştirme yöntemleri, geleneksel yazılım geliştirme yöntemlerine rakip olarak, 2001 yılında bir grup araştırmacı tarafından önerilmiştir (Agile Alliance, 2001). Geleneksel yazılım geliştirme yöntemlerine alternatif olarak geliştirilen çevik yazılım geliştirme yöntemleri düşük maliyet, yüksek üretkenlik ve kalite gibi özellikleri ile bir çok günümüzde bir çok organizasyon tarafından kullanılmaktadır (Agile Alliance, 2001). Agile manifesto ise 2001 yılında yinelemeli ve çevik metotlar ile ilgilenen 17 kişilik bir grubun ortak bir zemin bulmak için bir araya gelmesi ile oluşturulmuştur.

Çalışmalar sonucunda; bireyler ve etkileşimlere, araç ve süreçlerden; çalışan yazılıma, kapsamlı dokümantasyondan; müşteri ile işbirliğine, sözleşme pazarlıklarından; değişime karşılık vermeye, bir plana bağlı kalmaktan daha fazla değer verdiklerini belirttiler (Agile Alliance, 2001; Larman, 2004).

Çevik yazılım geliştirmenin genel yapısı yinelemeli yöntem üzerine kuruludur. Proje takımı her yineleme sonunda, müşteriye çalışan bir sistem gösterebilmeye odaklanmaktadır. Bu şekilde müşterinin taleplerini somut bir şekilde görebilmesi, istemediği durumlarda proje daha fazla ilerlenmeden müdahale edebilmesi sağlanmaktadır. Çevik yazılımın bu özelliği ile gereksiz zaman harcama durumu ortadan kaldırılarak, müşterinin yazılım geliştirme esnasında aktif rol aldığı esnek yazılım projeleri ile verimliliğin ve yazılım kalitesinin arttırılması hedeflenmektedir (Agile Manifesto, 2001; Akbayır ve diğ., 2010; Livermore, 2007).

1.4.1. Çevik yazılım geliştirme modelleri

Çevik yazılım geliştirme yöntemleri geleneksel yazılım geliştirme yöntemlerine bir alternatif olarak 2001 yılında bir grup araştırmacı tarafından geliştirilmiştir (Agile Alliance, 2001) ve son 10-15 yıldır araştırmacılar tarafından ilgi görmektedir (Dyba ve Dingsoyr, 2008). Farklı çevik yazılım geliştirme yöntemleri aşağıda sıralanmaktadır (Highsmith, 2002). Bunlar arasında en yaygın kullanılan yöntemler Scrum ve XP olarak olarak izlenmektedir (Schindler, 2008).

- Uç programlama (Extreme programming - XP)
- Çevik birleştirilmiş süreç (Agile unified process)
- Özellik güdümlü geliştirme (Future driven development - FDD)
- LEAN yazılım geliştirme (LEAN development)
- Dinamik sistem geliştirme modeli (Dynamic system development methodology - DSDM)
- Microsoft çözüm çerçevesi (Microsoft solution framework - MSF)
- Scrum

1.4.1.1. Uç programlama (Extreme programming - XP)

İşbirliğini, hızlı ve erken yazılımı ve yetenekli geliştirme pratiklerini vurgulayan XP; iyi bilenen çevik metotlar arasında yer almaktadır. 4 ana değer üzerine kuruludur. Bunlar; iletişim, sadelik, geribildirim ve cesarettir (Larman, 2004).

1.4.1.2. Çevik birleştirilmiş süreç (Agile unified process)

Rasyonel birleşik süreç (RUP) olarak bilinen bu yöntem, popüler yinelemeli süreçlerden biridir. Kısa zamanlı yinelemeler ile yüksek riskli ve yüksek değerli unsurlar geliştirme, müşteri değerini önemseme, projedeki erken değişikliklere uyum sağlama ve takımın birlikte çalışması prensiplerine dayanmaktadır (Larman, 2004).

1.4.1.3. Özellik güdümlü geliştirme (Future driven development - FDD)

Özellik güdümlü geliştirme, değişik boyutlara büyüyebilen, yinelemeli bir yazılım geliştirme sürecidir. İçerdiği beş süreç; bir modeli bütünüyle geliştirme, özellik listesini oluşturma, özelliklere göre plan, özelliklere göre tasarım ve özelliğin gerçekleştirilmesidir (Sütçü, 2003; Akbayır ve diğ., 2010).

1.4.1.4. LEAN yazılım geliştirme (LEAN development)

LEAN yazılım geliştirme, 2000'li yıllarda kullanılmaya başlayan bir çevik yazılım geliştirme türüdür. Geleneksel yazılım geliştirme metotlarının aksine, değişimi yazılım sürecine adapte ederek, yazılım üretmeyi hedefleyen bir yapıya sahiptir. LEAN yazılım geliştirmenin 7 prensibi boş harcamaların elenmesi, kalitenin sağlanması, bilginin oluşturulması, kararların ertelenmesi, hızlı dağıtım yapılması, takım üyelerinin birbirine karşı saygısı, tüm sürecin optimize edilmesidir (Highsmith, 2002; Poppendieck M. ve Poppendieck T., 2003).

1.4.1.5. Dinamik sistem geliştirme modeli (Dynamic system development methodology - DSDM)

1990'lı yılların ortalarında geliştirilen Dinamik sistem geliştirme modelinde amaçlanan yazılım geliştirme süreci içerisinde zamanı tutumlu kullanabilmektir. Dinamik sistem geliştirme modelinde yer alan 5 safha; fizibilite çalışması, iş çalışması, fonksiyonel model, tasarım - inşa ve kodlama safhalarıdır (Highsmith, 2002).

1.4.1.6. Microsoft çözüm çerçevesi (Microsoft solution framework - MSF)

İlk versiyonu 1993 yılında Microsoft tarafından çıkarılan Microsoft çözüm çerçevesi süreç geliştirmede takım modeli ve süreç modeli olarak iki model önermektedir.

Takım modeli çalışanları gruplandırarak her gruba bazı görevler verilmesini önermekte; süreç modeli ise gelişim açısından sürecin doğruluğunu ve uygunluğunu kontrol etmektedir (Turner, 2006; Akbayır ve diğ., 2010)

1.5. Çevik ve Geleneksel Yazılım Geliştirme Süreçleri Farkları

Geleneksel yazılım geliştirme modellerinin yerini almaya başlayan çevik yazılım geliştirme modellerine olan ilgi giderek artmaktadır. Bir çok organizasyon dokümantasyon fazlalığı, zamanında tamamlanamayan yazılım projelerinde yaşanan başarısızlıklar veya müşteriye teslim edilen yazılımlar ile ilgili alınan olumsuz geri bildirimler sebebi ile çevik yazılım geliştirme yöntemlerinden birine geçiş yapmaktadır. Geleneksel yazılım geliştirme süreçlerinde yer alan müşteri gereksinimlerinin projenin ilk aşamasında netleştirilerek daha sonra istenen değişikliklerin proje dışında bırakılması, tüm yazılım safhalarının art arda ilerlemesi, her safha sonunda belgeleme yapılması ve en önemlisi müşterinin gereksinim belirleme safhası haricindeki tüm yazılım aşamalarının dışında tutulması gibi sebepler, bir çok organizasyon için çevik yazılım geliştirmeye geçişin temel sebeplerini oluşturmaktadır (Bernal ve diğ., 2013; Başar ve diğ., 2015). Geleneksel yazılım geliştirme yöntemleri sadece dıştan gelen belirsizliklere cevap vermek için tasarlanmıştır, geliştirme ortamı döngünün başında yer almaktadır. Bu türlü yöntemlerde proje başladıktan sonra değişen ihtiyaçlara cevap verebilme özellikleri sınırlıdır. Scrum metodolojisi ise olabildiğince esnek olarak tasarlanmıştır (Schwaber, 1995). Tablo 2.1'de organizasyonların geleneksel yazılım geliştirme süreçleri yerine çevik yazılım geliştirme süreçlerini tercih etmelerine sebep olan önemli farklar karşılaştırmalı olarak gösterilmektedir.

Tablo 1.2. Çevik ve geleneksel yazılım geliştirme arasındaki farklar (Beck ve diğ., 2001; Bernal ve diğ., 2013; Pressman, 2005; Sommerville,2000)

Çevik yazılım geliştirme süreci	Geleneksel yazılım geliştirme süreci
Yeterli dokümantasyon	Dokümantasyon fazlalığı
Birlikte çalışma odaklı	Her bireyin kendi ortamında çalışması
Her yinelemede çalışan yazılım	Yazılım çalışmasını son aşamada test
Gereksinim değişikliklerine pozitif bakış	Gereksinimleri ilk aşamada belirleme
Yüzyüze iletişime önem verilmesi	Telefon, e-mail ve benzeri haberleşme
Takım içerisinde görev ayrımı yapılmaması	Analist, testçi ve yazılımcının belli olması

1.6. Çevik Yöntemlerin Yazılım Organizasyonlarında Uygulanma Durumu

Bu bölümde geleneksel yöntemlere alternatif olarak geliştirilen çevik yazılım geliştirme yöntemlerinin organizasyonlarda uygulanma durumları ile ilgili bilgiler verilmektedir. Çevik yöntemler ile ilgili Türkiye'de ve dünyada yapılan çalışmalar incelenerek, alan yazında yer alan bu çalışmalara ait sonuçlar raporlanmış, bu şekilde uluslararası düzeyde çevik yöntemlere bakış hakkında bilgi edinmek istenmiştir.

Çevik yazılım ile geliştirilen projeler ve bu konu ile ilgili yapılan anket çalışmaları izlendiğinde, çevik yazılım geliştirme metotlarının esneklik ve verimliliği yükselttiği gözlemlenmektedir (Schwaber, 1995; Baytam ve Kalıpsız, 2011). Bu çalışmalarda ayrıca, Çevik yöntemlerin adaptasyonunda bireysel ilginin etkili olduğu bildirilerek (Turk ve Vijayasarathy, 2008), en çok kullanılan çevik yazılım geliştirme yönteminin ise Scrum yazılım geliştirme yöntemi olduğu belirtilmiştir (Azizyan ve diğ., 2011; URL-3, 2014; Warma, 2012). Çevik yazılım geliştirme yöntemlerine yönelik ülkemizde ve yabancı ülkelerde gerçekleştirilmiş çeşitli büyüklükte çalışmalar arasından derlenen Çevik yazılım geliştirme ile ilgili yapılan anketlerin özeti, anketi gerçekleştiren araştırmacı bilgisi, gerçekleştirildiği bölge, gerçekleştirildiği yıl ve hedeflerini özetleyen bilgiler Tablo 2.2'de belirtilmektedir. Tablo 2.2'de izlenilen çalışmalar genel olarak 2006 ve 2014 yılları arasında gerçekleştirilmiştir.

Tablo 1.3. Çevik yazılım geliştirme ile ilgili çalışmaların özeti

Araştırma Referansı	Bölge / Anketi yapan	Yıl	Katılımcı sayısı	Hedef/Odak alanı
(Preuss, 2006)	Çeşitli ülkeler Digital Focus Firması	2006	128 organizayon - 136 yönetici	Çevik yaklaşımlar ve kullanımları önündeki engeller
(Turk ve Vijayasaraty, 2008)	Çeşitli ülkeler Turk ve Vijayasarathy	2008	98 katılımcı	Çevik yazılım geliştirme adaptasyonu ve kullanımı
(Abrahamsson ve Salo, 2008)	Avrupa Salo ve Abrahamsson	2008	13 yazılım organizasyonu	Çevik yöntemlerin durumu - XP, SCRUM
(Schindler, 2008)	Avusturya Schindler	2008	Raporlanmadı	Çevik yazılım geliştirme yöntem ve pratiklerinin durumu
(Azizyan ve diğ., 2011)	35 farklı ülke Azizyan, Magarian, Kajko-Mattson	2011	120 yazılım organizasyonu-121 katılımcı	Çevik yaklaşım yöntemleri ve birlikte kullanılan araçlar

Tablo 1.3. (Devam) Çevik yazılım geliştirme ile ilgili çalışmaların özeti

(VersionOne, 2013)	Raporlanmadı VersionOne	2013	Raporlanmadı	Çevik yöntemler ve yazılım geliştirmenin durumu
(Warma, 2012)	Hollanda, Endonezya Warma	2012	279 katılımcı	Çevik yazılım geliştirme başarısı
(Bustard ve diğ., 2013)	Kuzey İrlanda Bustard,Wilkie,Greer	2010 - 2012	24 katılımcı + 30 katılımcı	Çevik yazılım geliştirme prensip ve pratiklerinin uygulanma değişiklikleri
(Ambler, 2010)	Çeşitli ülkeler Ambler	2010	108 katılımcı	Çevik projelerin başarı durumu
(Ambler, 2014)	Çeşitli ülkeler Ambler	2014	114 katılımcı	Çevik yazılım geliştirme adaptasyonu
(Baytam ve Kalıpsız, 2011)	Türkiye Baytam, Kalıpsız	2011	30 katılımcı	Çevik yöntemler-SCRUM
(Agile Türkiye, 2013)	Türkiye Agile Türkiye	2013	540 katılımcı	Yazılım üretkenlik Raporu

Digital Focus (Preuss, 2006) firmasının 2006 yılında çeşitli ülkelerin 128 organizasyondan 136 yönetici ile gerçekleştirdiği çalışmanın sonuçlarına göre çalışmaya katılan organizasyonların % 81'i çevik yaklaşımları ya kullanmakta ya da kullanmaya başlamayı düşünmektedirler. Katılımcılardan % 51'i ise çevik yaklaşımları uygulamanın önündeki engel olarak bilgi ve beceri eksikliğini göstermektedirler.

Vijayasarathy ve Turk (Turk ve Vijayasarathy, 2008) organizasyonlarda çevik yazılım geliştirme yöntemlerinin kullanımını ve bunların zorluklarını değerlendirdikleri çalışmalarında, çeşitli ülkelerden 98 katılımcı ile gerçekleştirdikleri çalışmada XP ve test-first en yaygın yazılım geliştirme yaklaşımı olarak belirtilirken Scrum dördüncü sırada gelmektedir. Katılımcılarının büyük çoğunluğu (%75) projelerinde çevik yöntemleri kullandıklarını ifade etmişlerdir. Çevik yöntemlerin adaptasyonunda bireysel ilginin, engelleyici faktör olarak ise kurumsal direnç ve yönetim ilgisizliğinin etkili olduğu belirlenmiştir. Çalışmada ayrıca geleneksel yazılım geliştirme yöntemlerinden, çevik yazılım geliştirme yöntemlerine geçiş sürecindeki adaptasyonda bireysel ilginin etkili olduğu, engelleyici faktör olarak ise kurumsal direnç ve yönetim ilgisizliğinin öne çıktığı belirtilmektedir.

Salo ve Abrahamsson (Abrahamsson ve Salo, 2008) gömülü yazılım geliştirme organizasyonlarındaki çevik yöntemlerden özellikle XP ve SCRUM'ın durumuna belirlemek için 13 yazılım organizasyonundan 35 proje kapsamında bir anket çalışması gerçekleştirmişlerdir. Çalışmanın sonucunda gömülü yazılım endüstrisinde de çevik yöntemlerin kullanımının gerçekleştiği ve bu yöntemlere karşı olumlu algının yaygınlaştığı gözlemlenmiştir.

Schindler'in (Schindler, 2008) Avusturya bilişim endüstrisindeki çevik yazılım geliştirme yöntem ile pratiklerinin durumunu ve özellikle eş-programlama çevik prensibini göz önünde bulundurarak yaptığı çalışmanın sonucunda, Avusturya'da çevik yöntemlere yönelik farkındalığın bulunduğu ve gelecek planları içinde çevik yöntemler ve özellikle eş programlamayı kullanmayı planladıkları sonucunu elde etmiştir. Çevik yaklaşımlardan en yaygın kullanılanlar, XP (%46) ve SCRUM (32,8) olarak izlenmektedir. Çevik yöntemlerin uygulanmasının önündeki engel olarak ise bilgi eksikliği, yönetimin desteklememesi ve zaman eksikliği konuları en önemli nedenler olarak ön plana çıkmaktadır.

Azizyan, Magarian ve Kajko-Mattson (Azizyan ve diğ., 2011) daha çok çevik yaklaşımlar ile kullanılan araç ihtiyacı ve kullanımına yönelik olarak 35 farklı ülkeden 120 farklı yazılım organizasyonundan topladıkları 121 cevabı analiz ettikleri çalışmalarında SCRUM (% 54) ve XP'nin (% 32) ile en yaygın kullanılan çevik yöntemler olduğu sonucunu elde etmişlerdir. En yaygın kullanılan araçlar ise klasik duvar-yapışkan kağıt (%26), standart ofis araçları (Excel, vb.) (%23) ve MS Project (%8) olarak izlenmektedir. Yardımcı araçların en büyük eksikliği diğer sistemler ile entegrasyonlarının olmaması olarak belirtilmiştir. İhtiyaç duyulan özellikler olarak ise raporlama ve var olan sistemler ile entegrasyon özelliklerinin olması en çok tercih edilen özellik olarak belirtilmiştir.

VersionOne (URL-3, 2014) firması 2007 yılından bu yana çevik geliştirmenin durumunu tespit etmeye yönelik araştırma raporları yayınlamaktadır. En son 2013 yılında yayınlanan raporunda çevik yöntemleri gelecek projelerde adapte etme oranı % 69'dan % 83'e çıkmış, çevik yöntemler içerisinde en yaygın kullanılan ise % 72 ile SCRUM olarak seçilmiştir. Katılımcıların yaklaşık % 90'ı çevik yöntemleri kullanmanın değişen öncelikleri yönetme becerilerini artırdığını, % 84'ü ise çevik

yöntemler ile sağlanan görünürlüğün faydalı olduğunu vurgulamışlardır. En yaygın olarak kullanılan standart ofis yazılımları, Microsoft Project ve VersionOne gibi yazılımlar olarak izlenirken, en çok tavsiye edilen VersionOne (%93) olarak ön plana çıkmaktadır.

Warma (Warma, 2012) çevik yazılım geliştirmenin başarısını tespit etmeye yönelik yapmış olduğu yüksek lisans çalışması kapsamında, Hollanda ve Endonezya'dan 279 katılımcı ile gerçekleştirdiği anket sonucuna göre çevik yöntemler konusundaki farkındalığın %79 oranında olduğunu tespit etmiş ve katılımcıların yarı yarıya çevik yazılım geliştirme projelerinde yer aldıklarını belirlemiştir. Çevik projelerde çalışan katılımcıların neredeyse %80'i iki yıldan az bir zamandan beri bunu deneyimlemektedirler. En yaygın olarak kullanılan çevik yazılım geliştirme yöntemi olarak %90 ile SCRUM birinci sırada gelmektedir. Anket katılımcıları tarafından çevik yazılım geliştirmenin yazılım üretkenliği ve kalitesini olumlu yönde etkilediği düşünülmektedir. %60'dan fazla katılımcı çevik projelerin başarı oranlarının %50'den fazla olduğunu belirtmektedirler.

Ambler (Ambler, 2010; Ambler, 2014) 2007 yılından bu yana çevik yazılım geliştirme adaptasyonu ve çevik projelerin başarı oranları konuları ile ilgili anketler yapmaktadır. 2010 yılında 108 katılımcı ile gerçekleştirilen çevik projelerin başarı durumunu tespit etmeye yönelik anket sonuçlarına göre, katılımcıların %62'si çevik projeleri planlanan zamana göre teslim etmenin, %34'ü ise (sistemin yüklenmek üzere) hazır olduğunda teslim etmenin önemli olduğunu belirtmişlerdir. Yatırımların iyi geri dönüş sağlamasının, yazılımların belirlenen bütçe içerisinde bitirilmesine göre %60 daha önemli olduğu ifade edilmiştir. Çevik projelerin adaptasyonu ile ilgili 2014 yılında 114 kişinin katılımı ile gerçekleştirilen ankete göre ise, katılımcıların yaklaşık %15'i çevik teknikleri benimsemeye başladıklarını, %26'sı takımlarının yarısından azında çevik teknikleri uyguladıklarını, %49'u ise takımlarının yarısından fazlasında çevik teknikleri uyguladıklarını belirtmişlerdir. Organizasyonların çevik adaptasyonlarındaki başarısını etkileyen faktörlerin önem sıralamasında ise en önemli faktör %95 ile insanlar, en önemli ikinci faktör %88 ile süreçler, en önemli üçüncü faktör ise yaklaşık % 89 ile araçlar olarak izlenmektedir.

Türkiye'de çevik yöntemlerden Scrum ile ilgili gerçekleştirilen bir anket çalışmasının sonuçlarına (Baytam ve Kalıpsız, 2011) göre organizasyonların bu yöntemi kullanmaya başlamaları henüz erken aşamalarda yer almakta ve projeler küçük takımlar halinde gerçekleştirilmektedir. Yaygın olarak kullanılan araçlar duvar-yapışkan kağıt ya da Excel gibi yazılımlar olurken, destek sağlayacak araçlarda olması istenen özellik olarak takip edilebilirlik, kullanılabilirlik ve esneklik ön planda listelenmektedir.

Güncel olarak Agile Türkiye (Agile Türkiye, 2013) tarafından 2013 yılında gerçekleştirilen Yazılım Üretkenlik Raporu sonuçlarına göre de Türkiye'deki projelerin % 64'ünde çevik yöntemler uygulanmaktadır. Çevik yöntemler henüz baskın yazılım geliştirme yöntemi olarak kabul edilmemesine rağmen en yaygın olarak kullanılan çevik yaklaşımın da Scrum ve türevleri olduğu bilgisi paylaşılmaktadır.

Ard arda sıralanan bu konuların tümü Çevik yazılım geliştirme yöntemlerinin olumlu etkilerini belirtmektedir. Ancak her alanda ve sistemde olduğu gibi Çevik yöntemlerinde kullanıcıları tarafından tespit edilen olumsuz etkileri bulunmaktadır. Çevik yöntemler, son 10-15 yıl içerisinde yazılım dünyasında sıkça anlatılmasına ve uygulanmasına rağmen, geleneksel olarak tanımladığımız yazılım geliştirme yöntemleri halen tam olarak ortadan kalkmamakta ve bir çok büyük organizasyonda kullanılmaya devam etmektedir. Bu durumu organizasyonlarda Çevik yöntemlerin olumsuz etkilerinin olduğu algısı ve geleneksel yöntemlerden çevik yöntemlere geçişteki adaptasyon sürecinin fazla olması (Turk ve Vijayasarathy, 2008) gibi nedenlerle özetleyebiliriz. Konu ile ilgili daha önce yapılan araştırma ve makaleler incelendiğinde, Çevik yöntemlerin uygulanmasının önündeki engel olarak bilgi eksikliği, üst yönetimin desteklememesi ve zaman kısıtlılığı gibi konuların başlıca nedenler olarak ön plana çıktığı gözlemlenmektedir (Schindler, 2008). Ayrıca bu yöntemin tüm organizasyonlarda uygulanmasını engelleyici faktör olarak da kurumsal direnç ve yönetim ilgisizliği faktörlerini gösterilmektedir (Turk ve Vijayasarathy, 2008).

2. SCRUM YAZILIM GELİŞTİRME METODOLOJİSİ

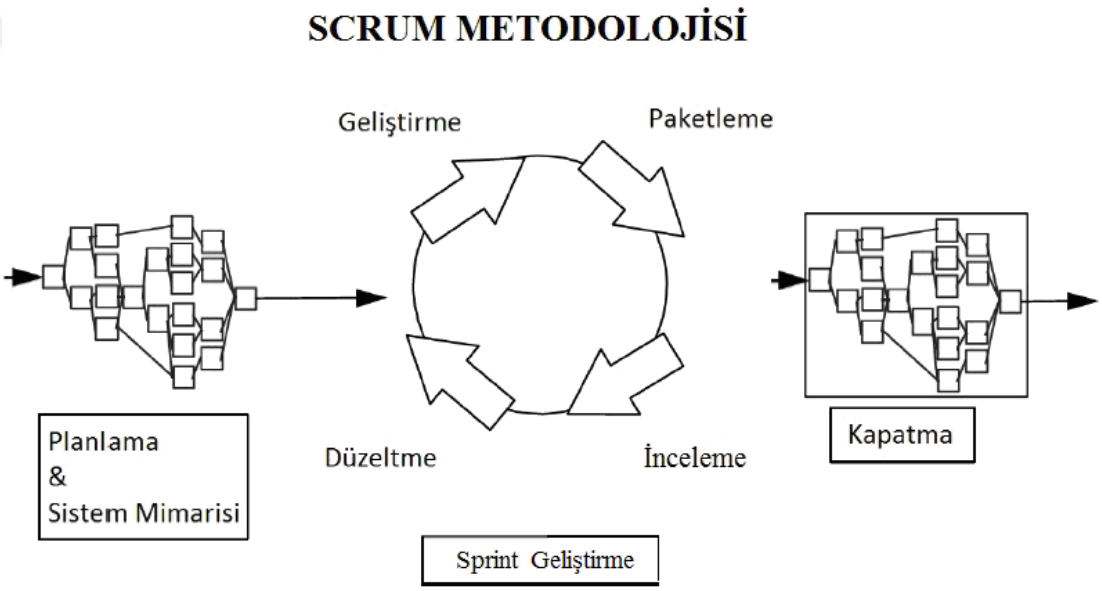
Bu bölümde bu tez çalışmasının ana odak noktasını oluşturan çevik yazılım geliştirme metodolojilerinden Scrum ile ilgili açıklamalar yapılmaktadır. Teorideki Scrum metodolojisinin, ana fazları, fazlar içerisinde gerçekleştirilen aktivitelerin neler olduğu, takip edilen süreç ve Scrum içerisinde takım üyelerinin aldıkları rol ve sorumluluklar anlatılmaktadır. Sonrasında Scrum temel kavramları olarak da adlandırılan Scrum pratikleri listelenmektedir. Son olarak da Scrum yazılım geliştirime metodolojisinin uygulanmasında karşılaşılan problemler ve zorluklar ilgili alan yazındaki örnekleri ile bahsedilmektedir.

Çevik yazılım geliştirme süreçleri içerisinde Scrum yazılım geliştirme yöntemi diğer süreçlere göre öne çıkmaktadır (Rising ve Janoff, 2000). Günümüzde bir çok organizasyonda çevik yazılım geliştirme süreçlerinde Scrum kullanılmaktadır (Pries-Heje, J. ve Pries-Heje, L., 2011). 1990'ların başlarında Ken Schwaber tarafından uygulanan ve Scrum adı verilen bu yaklaşım (Sutherland, 2004) yazılım endüstrisinde son 10-15 yıl süresince kabul gören, yinelemeli, nesne tabanlı geliştirme döngüsü içerisinde kullanılan, başarılı pratikler temeline dayandırılmış bir yazılım geliştirme sürecidir (Schwaber ve Sutherland, 2010; Schwaber, 1995).

Scrum küçük takımlar ile esnek bir zamanlama yapısı kullanarak çalışan, devamlı gözden geçirmeleri olan nesne tabanlı bir yazılım geliştirme modelidir (Baytam ve Kalıpsız, 2011; Schwaber, 1995). Scrum yazılım geliştirme yönteminde projeler büyüklüklerine göre değişen sayıda yinelemelere (Sprint) ayrılarak geliştirilmekte, farklı organizasyonlarda değişebilirlik göstermesiyle birlikte genel olarak her bir yineleme yaklaşık 4 hafta kadar sürebilmektedir. Scrum projelerinde kısa süreli proje yapmak esas alınmaktadır. Proje süresinin 1 yılı geçmemesi hedeflenir (Abrahamsson ve diğ., 2002). Bu zamanlamaya uyamayacak büyüklükte projeler ise fazlandırma yapılarak çalışılabilir.

2.1. Scrum Metodolojisi

Geleneksel yazılım süreçlerinde olduğu gibi Scrum yazılım geliştirme süreci içerisinde de çeşitli safhaların yer aldığı bir model yapısı bulunmaktadır. Scrum metodolojisi olarak adlandırılan bu model, genel olarak 3 safhadan oluşmaktadır. Bu safhalar Hazırlık, Sprint Geliştirme ve Kapatma (Bitiş) olarak tanımlanmaktadır (Välimäki ve Kääriäinen, 2008). Şekil 2.1'de izlendiği üzere Hazırlık Safhası, Planlama ve Sistem mimarisi hazırlanması; Sprint Geliştirme safhası Geliştirme, Paketleme, Düzeltme ve İnceleme aşamalarını; Kapatma safhası ise kullanıcı için hazırlanan eğitim materyalleri dağıtımı ve sistemin entegrasyonlarını içermektedir.

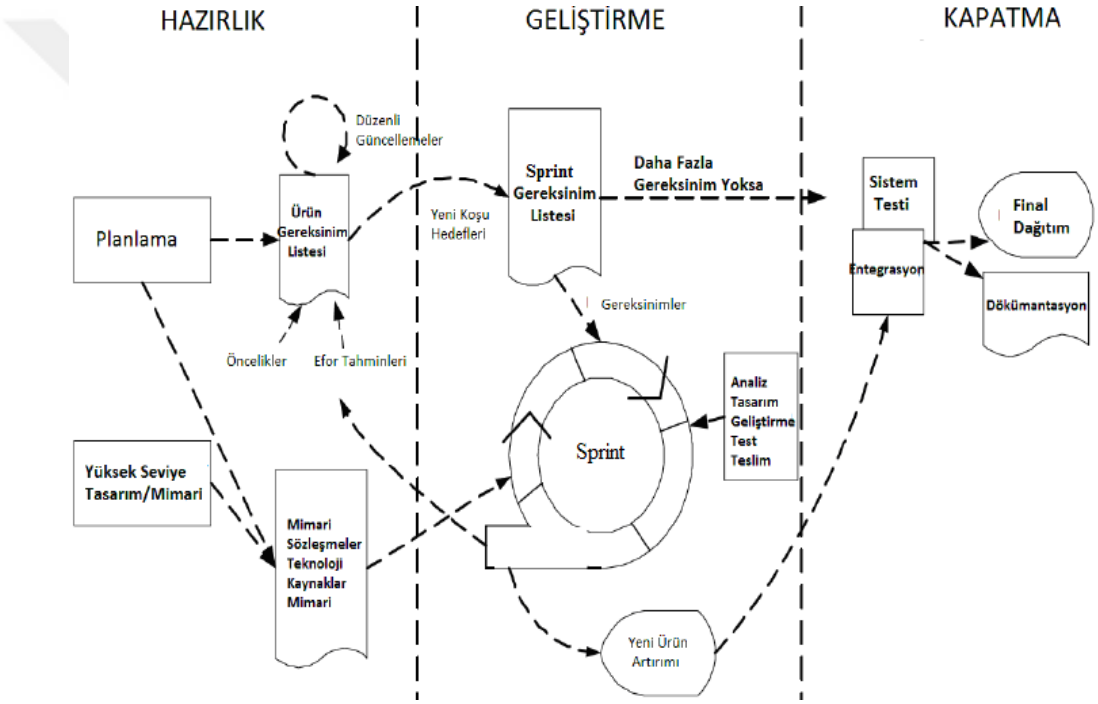


Şekil 2.1. Scrum Metodolojisi (Schwaber, 1995)

Scrum metodolojisi içerisinde yer alan ilk ve son fazlar süreç tanımını içermektedir, bu bölümde girişler ve çıkışlar tam olarak tanımlanır. Planlama aşamasında bazı yinelemeleri içermesi ile birlikte bu süreçlerde akış doğrusal olarak tanımlanır. Sprint fazı ise deneysel aşamadır, bir çok projede Sprint fazında süreçler kontrol edilemez ve tam olarak tanımlanamaz. Bu yapı dış kontrol gerektiren bir kara kutu yapısı olarak tanımlanabilir. Bu faz, esnekliği en yüksek derecede kullanırken, herhangi bir kaosu önlemek amaçlı kontroller ve risk yönetimi de içermektedir. Bu tanımlama ile Sprint fazının doğrusal olmadığı belirtilmektedir. Son ürünün geliştirildiği bu aşamada süreci oluşturmak için deneme yanılma yöntemi uygulanmaktadır. Planlama ve Sprint fazlarında her an değişen çevresel karmaşıklık, rekabet, zaman, kalite ve mali baskılara açık olan proje Kapanış aşamasında teslim edilir (Schwaber, 1997).

2.2. Scrum Fazları (Safhaları)

Hazırlık, Sprint Geliştirme ve Kapatma (Bitiş) safhalarından oluşan Scrum metodolojisi içerisindeki aşamalar aşağıda Şekil 2.2'de detaylı olarak görülmektedir. Şekil incelendiğinde, hazırlık safhası içerisinde, Ürün gereksinim listesi ve mimari sözleşmelerin hazırlanması; Sprint Geliştirme safhasında Sprint gereksinim listesinin hazırlanarak Analiz, Tasarım, Geliştirme, Test ve Teslim aşamalarından geçirilmesi ve son safha olan Kapatma (Bitiş) safhasında sistem testlerinin yapılarak son kullanıcı dokümanlarının dağıtılmasının gösterildiği izlenmektedir (Kääriäinen ve Välimäki, 2008).



Şekil 2.2. Scrum Fazları (Kääriäinen ve Välimäki, 2008; Abrahamsson ve diğ., 2002)

2.2.1. Hazırlık

Her Scrum fazının belirli basamakları bulunmaktadır. Hazırlık safhasının basamakları Planlama ve Mimari-Yüksek Seviye Tasarımdan oluşmaktadır (Schwaber, 1995).

2.2.1.1. Planlama

Yazılım geliştirmenin hazırlık safhasında yer alan ilk aşama planlama aşamasıdır. Bu aşamada mevcut ürün iş listesi üzerinde maliyet ve zaman belirleme işlemleri yapılmaktadır (Schwaber, 1995). Planlama aşamasında gerçekleştirilen aktiviteler aşağıdaki şekilde listelenebilir.

- Geniş kapsamlı gereksinim dokümanı (ürün iş listesi) geliştirme
- Teslim tarihi ve fonksiyonel özellikleri tanımlama (bir veya daha fazla sürüm için)
- Geliştirme için en uygun sürümün seçimi
- Yeni sürümü oluşturmak için proje takımını tanımlama
- Risk kontrollerini belirleme ve risk değerlendirme
- Geliştirme araçları ve altyapı doğrulama
- Seçilen sürümün ürün iş listesi için ürün paketlerini eşleştirme
- Ürün iş listesinin incelenmesi ve düzenlemelerin yapılması
- Geliştirme içeriği, materyal toplama, pazarlama, eğitim ve sunum gibi sürüm maliyetleri tahmini
- Yönetim onayı ve finansman doğrulama (Schwaber, 1995)

2.2.1.2. Mimari-yüksek seviye tasarım

Ürün iş listesinde yer alan adımların nasıl gerçekleştirileceği bu bölümde planlanır. Mimari-yüksek seviye tasarım aşaması sistem mimarisi modifikasyonlarını ve yüksek seviyeli tasarımı içermektedir (Schwaber, 1995). Mimari-Yüksek seviye tasarım aşamasında gerçekleştirilen aktiviteler aşağıdaki şekilde listelenebilir.

- Belirlenen ürün iş listesinin incelenmesi
- Ürün iş listesini uygulama için gerekli değişiklikleri tanımlama
- Yeni gereksinimleri desteklemek üzere sistem mimarisi tanımlanması
- Değişiklikleri uygulama veya geliştirme sırasında oluşan problemleri tanımlama
- Gözden geçirme toplantılarını belirleme, ürün iş listesindeki her adım için yaklaşım düzenleme
- Gerekli ölçüde alan analizi yapılması (Schwaber, 1995)

2.2.2. Sprint geliştirme

Scrum yazılım geliştirme yönteminde tekrarlamalar Sprint olarak adlandırılır. Bir Sprint süresi genellikle 2 ila 4 haftadır, ancak duruma göre daha uzun veya daha kısa olabilir (Durasiewicz ve diğ., 2009). Scrum ardarda sprint'lerden oluşan, proje yönetimi ve planlama yöntemlerine odaklı mühendislik detayları içermeyen yinelemeli bir yaklaşımdır (Pries-Heje, J. ve Pries-Heje, L., 2011; Schwaber, 1995; Güneş, 2011).

Sprint Geliştirme safhası geliştirilen işin yinelemeli döngüyü içeren aşamasıdır. Sprint Geliştirme safhasında, zaman, maliyet, rekabet, gereksinimler ve kalite değişkenleri göz önünde bulundurularak yeni sürümün (dağıtımın) fonksiyonelliği geliştirilir. Sprint tamamlandıktan sonra Kapatma safhasına geçilir (Schwaber, 1995). Temel olarak bu süreç içerisinde, Sprint planlama ve Sprint gözden geçirme toplantıları, ürün standartları düzenlemeleri ve sunuma hazır hale gelinceye kadar yinelenen Sprint'ler ile ürün geliştirilmesi yer almaktadır (Baytam ve Kalıpsız, 2011).

Geleneksel yazılım geliştirme süreçleri ile Scrum yazılım geliştirme sürecinin önemli farklarından biri Sprint aşamasında yer alan analiz, tasarım ve geliştirme süreçlerinin önceden bilinemiyor olmasıdır. Bu önceden bilinemeyen süreci yönetmek için bir kontrol mekanizması kullanılır ve risk bu şekilde kontrol altına alınmaya çalışılır. Sonuç ise esneklik, uyumluluk ve güvenilirlik olarak özetlenebilir (Schwaber, 1995).

Her Sprint geliştirme, paketleme, inceleme ve düzeltme alt aşamalarını içerir. İlk aşama olan geliştirme, Sprint aşamasında yer alan analiz, tasarım ve geliştirme süreçlerinin gerçekleştirildiği adımdır. Değişiklikleri tanımlama, kodlama için gereksinim listesinin içinde paketler, paketlerin açılması, alan analizi yapılması, tasarım, geliştirme, uygulama, test ve değişiklikleri belgelemelerini içermektedir. Bu aşamada aynı zamanda sürecin keşfedilmesi, bulunması ve uygulaması yer almaktadır. İkinci aşama olan paketleme, gereksinim listesinde yer alan maddelerin nasıl uygulanacağı, yürütülebilir versiyonun oluşturulması ve paketlerin kapatılması işlerinin yapıldığı adımdır. Üçüncü aşama olan inceleme adımında, tüm Scrum takımının toplanarak işi sunması ve süreci incelemesi, sorunların çözülmesi, yeni gereksinimlerin eklenmesi işlemleri gerçekleşmektedir. Bu bölümde ayrıca risk

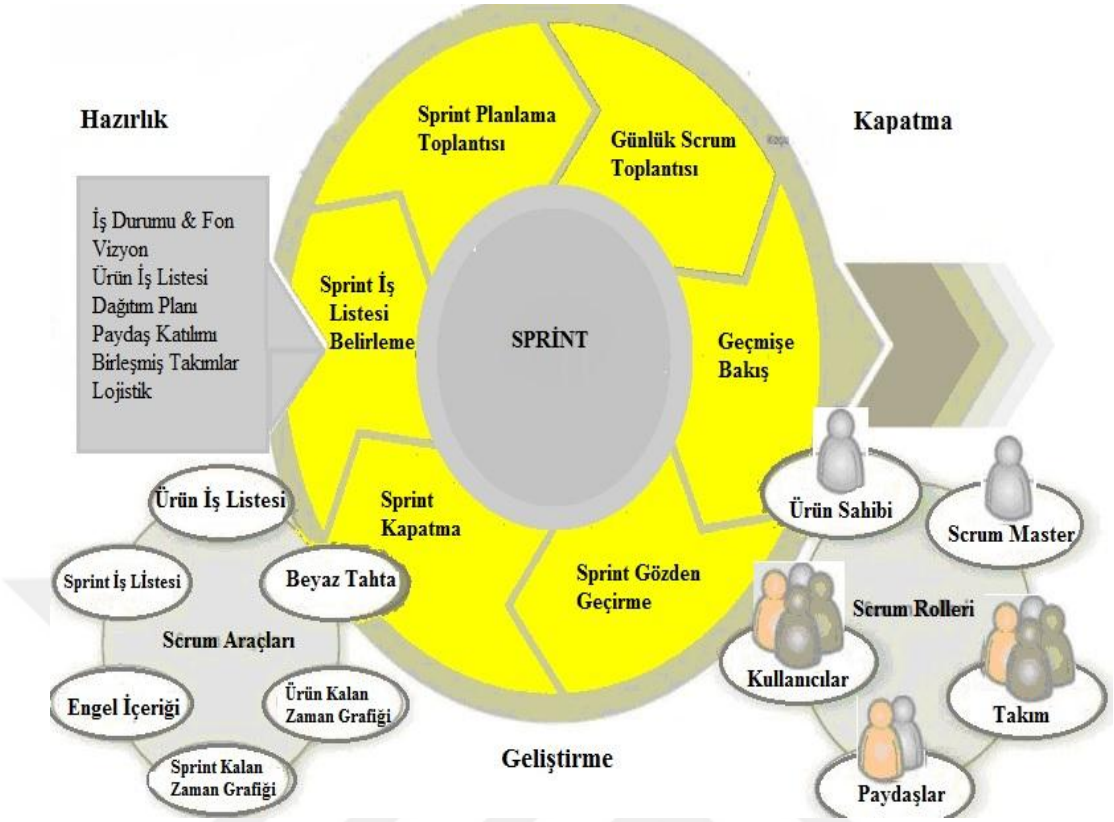
inceleme ve gerekli cevapların tanımlanması yapılmaktadır. Son aşama olan düzeltme adımında ise gözden geçirme toplantısında toplanan bilgiyi farklı bakış açısı ve yeni özellikler ile birleştirme işlemi gerçekleşmektedir (Schwaber, 1995).

2.2.3. Kapatma

Projenin tamamlandığı bu aşamada son dokümantasyonu içeren Sürüm (dağıtım) için hazırlık yapılması, test ve dağıtım aşamalarından oluşur. Genel sürüm için geliştirilen ürün hazırlanır. Kapatma aşaması içerisinde; entegrasyon, sistem testi, kullanıcı dokümanları, eğitim ve pazarlama materyallerinin hazırlanması yer almaktadır (Schwaber, 1995).

2.3. Scrum Süreci

Scrum içerisinde Sprint adı verilen tekrarlamalar içerisinde gerçekleştirilen aktiviteler Şekil 2.3'te izlenmektedir. Her bir Sprint; Hazırlık, Sprint Geliştirme ve Kapatma (Bitiş) safhalarını içermektedir. Bunlara ek olarak Sprint içerisinde yapılan işler Sprint iş listesi belirleme, Sprint planlama toplantısı, günlük Scrum toplantısı, Sprint gözden geçirme, geçmişe bakış ve Sprint kapatma olarak listelenmektedir. Hazırlık aşamasının da detaylı olarak gösterildiği şekilde, bu safha yapılacak işin durumu, işin yapılması için ayrılacak bütçe, vizyon, ürün iş listesinin belirlenmesi, Kapatma aşaması sonrası yapılacak ürün dağıtım planı, paydaşların katılımı, Scrum ile ilgili çalışan takımların birleşimi ve lojistik gibi süreç içerisinde yer alması muhtemel her konunun görüşüldüğü ve bunlarla ilgili planların yapıldığı aşama olarak izlenmektedir.



Şekil 2.3. Genel Scrum süreç yapısı (URL-1'den alıntılayan Baytam ve Kalıpsız, 2011)

2.4. Scrum Rolleri

Çevik yazılım geliştirmenin en önemli özelliklerinden biri proje ile ilgili her kişinin proje takımında aktif rol almasını sağlayarak, onları sürece dahil etmesi ve ortak bir hedefe doğru hareket eden takımı oluşturma çabasıdır. Şekil 3.3'de görülebildiği üzere Scrum süreci içerisinde yer alan Scrum rolleri, ürün sahibi, Scrum master, kullanıcılar, Scrum takımı ve varsa paydaşlar olarak listelenmektedir. Scrum içinde tüm takım üyelerinin aynı odada çalışması özelliği takım üyelerinin etkileşimli olarak çalışabilmesini sağlarken, zaman tasarrufu açısından da faydalı olmaktadır.

Teorideki Scrum modeli uygulanırken belirlenen roller arasında proje lideri rolü yer almamaktadır. Ekip üyelerinin bir yöneticisi bulunmazken, sürecin yönetimi ise Scrum master'a bırakılmıştır (Hundermark, 2009). Scrum takımı içerisinde görev alan her takım üyesi analiz, yazılım ve test gibi yazılım safhalarının tümünde görev alabilen sürekli iletişim halindeki olan çalışanlardan oluşur. Ürün sahibi (Product Owner), Scrum takım üyesi (Scrum Team Member) ve Scrum yöneticisi (Scrum Master) olmak üzere temelde 3 rol bulunur (Cohen ve diğ., 2004; Başar ve diğ.,

2015). Çevik projelerde ürün sahibi ile proje ekibinin beraber çalışması, bu projeleri geleneksel yazılım geliştirme yöntemleri ile çalışan takımlardan ayıran özelliklerden biridir. Bu şekilde takımda etkileşim artmakta, her Sprint sonunda çalışan bir yazılım ürün sahibine sunulabilmektedir (Abrahamsson ve diğ., 2002).

2.4.1. Ürün sahibi (Product owner)

Çevik yazılım geliştirmenin en önemli unsurlarından biri müşteri memnuniyetidir. Çevik yazılım geliştirme kullanılarak gerçekleştirilecek projelerde, müşteri gereksinimleri ve kısıtları büyük önem taşımaktadır. Scrum'da müşteri ile Scrum proje ekibi arasındaki etkileşimi sağlayan ürün sahibidir. Scrum proje takımında, müşteri tarafında çalışan, müşterinin isteklerini anlayan ve bunu proje ekibine anlatan kişiye ürün sahibi denilmektedir. Ürün sahibinin etkisi ile projenin müşteri ile birlikte geliştirilmesi ve gereksinimlerin doğru şekilde anlaşılması hedeflenmektedir. Ürün sahibi Scrum proje takımı içerisinde aktif olarak rol almaktadır (Akbayır ve diğ., 2010).

Aynı zamanda müşteri tarafından proje başında belirlenen ürün sahibi ürünü yönetmekle yükümlüdür ve şeffaf olmak zorundadır. Ürün sahibi, ürün iş listesinde yer alacak işler ile ilgili gereksinimleri ifade eden ve bunların doğru olarak, müşterinin isteğine uygun şekilde belirlenmesini sağlayan kişidir (Baytam ve Kalıpsız, 2011). Ayrıca kullanıcı hikâyelerini belirler; takım yazılım yaşam döngüsü içinde yer alan faaliyetleri gerçekleştirir (Cohen ve diğ., 2004).

2.4.2. Scrum yöneticisi (Scrum Master)

Scrum yöneticisi Scrum takımını organize eder ve ürün sahibine yardımcı olur (Cohen ve diğ., 2004). Takımın Scrum pratikleri uyumunu kontrol etmekle yükümlüdür. Scrum yöneticisinin görevlerini, takımı yönlendirme, üretkenliği arttırma, fonksiyonlar arası etkileşimi sağlayarak, engelleri kaldırma, dış etkenlere karşı takımı koruma ve Scrum sürecinin doğru işlemesini sağlama olarak detaylandırabiliriz (Schwaber ve Sutherland, 2007).

2.4.3. Takım üyesi (Team Member)

Scrum Takım üyeleri, geliştiriciler, testçiler, analistler, tasarımcılar ve kullanıcılardan oluşabilir (Schwaber ve Sutherland, 2007). Takım kendi kendini yönetir, genellikle herkes her işi yapabilir, takımın asli görevi her sprint sonunda çalışan bir yazılımı Ürün sahibine sunmaktır. Takım üyeleri arasında üstlük ilişkisi yoktur (Baytam ve Kalıpsız, 2011). Özellikle yinelemeler sırasında Scrum takımı içerisinden üye çıkarılmamalı veya eklenmemelidir. Ekip, çalışma disiplini olan, motivasyonu yüksek, iletişimi kuvvetli, olumlu düşünce yapısına sahip, kararlı kişilerden oluşturulmalıdır (Akbayır ve diğ., 2010).

2.5. Scrum Pratikleri

Scrum pratikleri Scrum yazılım geliştirme yönteminin temel kavramları arasında yer almaktadır. Bu pratikler; ürün iş listesi (Product backlog), Sprint planlama (Sprint planning), Sprint planlama toplantıları (Sprint planning meeting), Sprint iş listesi (Sprint backlog), günlük Scrum toplantıları (Daily Scrum meeting), Sprint gözden geçirme toplantıları (Sprint review meeting) olarak tanımlanmaktadır (Beedle ve Schwaber, 2001; Beedle ve Schwaber, 2002; Abrahamsson ve diğ., 2002).

Scrum'da, ürün iş listesi ile öncelikler belirlenmekte ve her sprint sonunda ürün sahibi tarafından müşteriye sunum yapılmaktadır. Ürün iş listesi geliştirilecek olan maddelerin listesini içermekte, işlevsellik ise kullanıcı hikayeleri (User stories) ile sağlanmaktadır. Her Sprint'e başlarken öncelikle Sprint planlama toplantısı yapılmaktadır. Ürün sahibi, Sprint planlama toplantısında, ürün iş listesi içerisinden o Sprint içerisinde yapılacak maddeleri seçerek, Sprint iş listesini oluşturur. Bu toplantılar görünürlük sağlayarak, takım üyelerine Sprint ile ilgili planlama yapabilme fırsatı vermektedir. Her Sprint (yinemele) sonunda ise müşteriye çalışan yazılımın bir parçası sunulur. Bir Scrum projesi birden fazla yineleme'den oluşur. Ayrıca proje takımı tarafından her gün 15 dakikalık günlük Scrum toplantısı yapılmaktadır. Toplantı süreç sorumlusu olarak bilinen Scrum master tarafından yönetilmektedir. Günlük Scrum toplantıları, beyaz tahta önünde ayakta olarak yapılmaktadır. Toplantı esnasında her takım üyesi; "Dün ne yaptım? Bugün ne yapacağım? Karşılaşılan problemler ve yenilikler nelerdir?" sorularına cevap aramaktadır. Sprint gözden geçirme toplantıları ise her Sprint sonunda yapılan ve

geliştirilen işlevselliğin tüm bölümlere gösterildiği toplantılardır. İçeride ve dışarıda görev alan proje çalışanlarının katılımı ile gerçekleştirilir. Son olarak Sprint sonunda proje takımı tarafından geçmişe bakış yapılmakta ve "Sprint sırasında iyi olanlar neler, kötü olanlar neler ve gelecek sprint için ne gibi değişiklikler yapılmalıdır" sorularına cevap aranmaktadır (Pries-Heje, J. ve Pries-Heje, L., 2011; Durasiewicz ve diğ., 2009).

2.5.1. Sprint planlama (Sprint planning)

Her Sprint başında yapılan Sprint planlama toplantısında, ürün sahibi ve Scrum takım üyeleri tarafından ürün iş listesi kontrol edilmektedir. Ürün sahibi ile ortak karar alan takım üyeleri, ürün iş listesi içersinden mevcut Sprint içerisinde gerçekleştirilecek işlerin seçimini yaparlar. Bu seçim sırasında işlerin öncelikleri ve önemleri göz önüne alınır. Ürün sahibi tarafından belirtilen ürün iş listesi içerisinden, Sprint içine dahil olacak işler bu şekilde belirlenir. Bu yapı Scrum yazılım geliştirme sürecinin temel özelliklerinden biridir (Schwaber ve Sutherland, 2007).

2.5.2. Sprint gözden geçirme (Sprint review)

Sprint gözden geçirme toplantıları 4 haftalık Sprint süreci sonunda yapılır. Bu toplantıda Sprint esnasında geliştirilen ürünün sunumu yapılmaktadır. Bu toplantı için ayrı bir sunum yapılması gerekliliği bulunmamaktadır, takımın Sprint içerisinde gerçekleştirdiği ürünün gösterilmesi yeterlidir. Toplantıya Scrum takım üyeleri, Scrum master ve ürün sahibi dışında, konu ile ilgili uzmanlar, kullanıcılar ve paydaşlar da katılabilir (Schwaber ve Sutherland, 2007).

2.5.3. Günlük Scrum toplantısı (Daily scrum)

Her iş günü yapılan ve tüm Scrum takım üyelerinin katıldığı toplantılar, günlük Scrum toplantısı olarak adlandırılmaktadır. Süresi 15 dakika ile kısıtlanan bu toplantılar, tüm takım üyeleri tarafından cevaplanan 3 soru üzerine kuruludur (Schwaber ve Sutherland, 2007).

Bu sorular;

- Dün ne yaptım?
- Bugün ne yapacağım?

- İşimi yaparken karşılaştığım sorunlar ve engeller nedir?

Scrum master'ın görevi bu toplantılar içerisinde not tutarak, toplantı sonrasında sorunları takım üyeleri ile bire bir çözmektir. Takım iletişimi açısından faydalı olan bu toplantılar 15 dakikalık süreyi aşmamalı ve belirlenen amaç dışına çıkmamalıdır (Schwaber ve Sutherland, 2007).

2.5.4. Geçmişe bakış (Retrospective)

Geçmişe bakış toplantıları günümüzde bir çok Scrum proje takımı tarafından uygulanmamaktadır. Bu toplantılarda Scrum takım üyeleri tarafından neyin çalışıp neyin çalışmadığı ve gelecek sprint için ne gibi değişiklikler yapılması gerektiği sorgulanmaktadır (Schwaber ve Sutherland, 2007; Pries-Heje, J. ve Pries-Heje, L., 2011; Durasiewicz ve diğ., 2009).

2.5.5. Ürün gereksinim listesi (Product backlog)

Scrum'da gereksinimler müşterilerden alınarak kullanıcı hikayeleri (User stories) oluşturulur. Müşteri ile anlaşılarak belirlenen gereksinimlerin oluşturduğu listeye ise ürün gereksinim listesi (Product backlog) denir. Scrum bu şekilde zaman kutulu (Time boxed) olarak tanımlanır (Cohen ve diğ., 2004; Başar ve diğ., 2015).

Scrum takımı projeye başlarken öncelikle, proje içerisinde yapılacak tüm işler bir listeye kayıt edilmekte ve bu listeye ürün iş listesi denilmektedir. Scrum yazılım geliştirme ekibinin öncelikli görevi daha önce ürün iş listesinde belirlenen gereksinimleri tam olarak karşılayan yazılımı geliştirebilmektir (Akbayır ve diğ., 2010).

Ürün gereksinim listesinde yer alan gereksinimlerin önceliklendirme işlemi ürün sahibi tarafından yapılmaktadır. Ürün gereksinim listesi içerisinde yer alan gereksinimlere yenileri eklenebilir veya gereksizler çıkarılabilir (Hundermark, 2009).

2.5.6. Sprint listesi (Sprint backlog)

Her Sprint başında yapılan Sprint planlama toplantısında 4 hafta sürecek Sprint içerisine dahil edilecek işler belirlenmektedir. Oluşturulan bu liste Sprint iş listesi olarak tanımlanmaktadır (Akbayır ve diğ., 2010). Ürün gereksinim dokümanı

içerisinde yer alan maddeler içerisinden bir kısmı 4 hafta süreli yineleme içerisinde kullanılmak üzere Sprint listesi içerisine kayıt edilir. Sprint listesinde yer alan gereksinimlerin ürün gereksinim listesinde de bulunması gerekmektedir (Hundermark, 2009). Sprint listesindeki işler için efor tahmini bu aşamada yapılmaktadır. Takım üyeleri Sprint listesi içerisinde yer alan gereksinimleri üzerlerine alırlar, ayrıca bu liste üzerinde ekleme, çıkarma ve silme işlemlerini yapabilirler. Sprint süresince çalışılan işler için süre tahmini güncellemeleri yapılır (Baytam ve Kalıpsız, 2011).

2.5.7. Sprint kalan zaman grafiği (Burndown chart)

4 haftalık Sprint süresi içerisinde Sprint dokümanı içerisinde yer alan işlerin ne kadarının yapıldığı ve hedeflenen olarak ne kadar yapılması gerektiğini karşılaştırmalı olarak gösteren grafiğe "Sprint kalan zaman grafiği" denilmektedir. Scrum takımı içerisinde görev alan kişiler günlük olarak gerçekleştirebilecekleri iş bilgisini girerler. Daha sonra Sprint içerisinde bu işlerin gerçekleşme durumları bu grafik ile takip edilebilir (Schwaber, 2004). Bu grafik Sprint sonunda, Scrum takımının taahhüt ettiği gereksinimleri gerçekleştirme durumunun izlenmesi açısından fayda sağlamaktadır (Hundermark, 2009).

2.5.8. Ürün kalan zaman grafiği (Burndown chart)

Ürün kalan zaman grafiği, ürün iş listesinde yer alan tüm işlerin ne kadar süre ile yapılacağının tahminleri ile hesaplanır. Ürün iş listesi tahminleri, oluşturuldukları zaman yapılır. Ürün iş listesinde yer alan işlere proje boyunca ekleme yapılabilir ve listede yer alan işler istenildiği zaman güncellenebilir (Schwaber ve Sutherland, 2010).

2.6. Scrum Problemleri

Yazılım organizasyonlarında son dönemlerde oldukça sık kullanılmaya başlanan çevik yöntemlerin, geleneksel yöntemlere oranla daha kullanılabilir olduğu ve ortaya çıkan yazılımlar ile yazılım geliştirme sürecindeki kaliteyi arttıklarına yönelik bir çok çalışma olmasına rağmen geleneksel yöntemler büyük ve orta ölçekli organizasyonlarda halen kullanılmaya devam etmektedir. Bunun sebebi kimi

organizasyonlarda alışkanlıklara olan eğilimin devam etmesi olarak gösterilirken bir çok organizasyon adaptasyon sürecinde yaşadıkları sorunlar sebebi ile tam anlamıyla çevik yöntemlere geçişi sağlayamadıklarını belirtmektedir (Schindler, 2008; Turk ve Vijayasarathy, 2008). Çevik yöntemlerin halen tüm yazılım organizasyonlarda uygulanmamasının diğer sebepleri arasında ise kurumların gerek personel gerekse müşteri tarafından yeni yönteme gösterdikleri direnç ve üst yönetimin konuya karşı olan ilgisizliği gibi etkenler rol oynamaktadır (Turk ve Vijayasarathy, 2008). Konu ile ilgili farklı çalışmalar incelendiğinde, çevik yöntemlerin uygulanmasının önündeki bir diğer engel olarak bilgi eksikliği, bu eksikliği tamamlamak üzere harcanması gereken zamanın kısıtlılığı ve üst yönetimin bu türlü kapsamlı değişikliklere sıcak bakmaması gibi konular başlıca nedenler olarak gösterilebilir (Schindler, 2008). Yazılım dünyasında sürekli değişen ihtiyaçlar sebebi ile takımların aşırı çalışma gerekliliği, yönetim tarafından gelen hedef ve tarih baskısının takıma olan olumsuz etkileri ve müşteri ile bire bir iletişimin kısıtlı olduğu zamanlarda yaşanabilecek zorluklar da organizasyonların çevik yöntemlere geçiş sürecinde yaşadıkları olumsuzluklar açısından önemli olarak görülmektedir (Balçiçek, 2016).

2.7. Scrum Ama ve Scrum Ve

Alanyazında Scrum ama kavramı, Scrum yazılım geliştirme yöntemine ait bazı pratik ve kavramları uyguladıklarını belirten organizasyonlar için kullanılmaktadır (Eloranta ve diğ., 2015; Sutherland, 2010). Organizasyonlarda Scrum pratikleri ve uygulamalarının tam olarak kabul edilmesi yönetim düzeyinde anlayış gerektiren bir durumdur. Organizasyonların Scrum'ı tam olarak anlamadan ve Scrum'a geçiş sonrasında oluşacak sonuçları öngörmeden hareket etmesi sonucunda ortaya çıkan bu durum Scrum yönteminin organizasyonlara göre modifiye edilmesine sebep olmuştur. Modifiye edilen bu yapı Scrum ama olarak tanımlanmaktadır. Organizasyonlar zaman zaman Scrum içerisinde yer almayan bazı pratikleri kullanarak iyi sonuçlara ulaşabilmişlerdir. Bu durum ise Scrum ve (ScrumAnd) olarak tanımlanmaktadır (Krishna & Basu, 2011). Özellikle Scrum yazılım geliştirme yöntemini organizasyonlarında yeni uygulamaya başlayanlar tarafından daha sık kullanılan Scrum ama kavramı, yıllardır Scrum uyguladıklarını belirten bir çok organizasyonda da kullanılmaktadır. Scrum'ın bazı pratiklerini uygulamada zorlanan organizasyonlar genellikle Scrum'ı esnetmeyi çözüm olarak görmekte, bazı

özelliklerini uygularken, bazılarını ise kullanmamaktadırlar. Scrum takımlarının %80'i bu türlü Scrum ama mantığı ile hareket etmektedir (Yitmen, 2015). 2015 yılında Scrum rollerinden Scrum master rolü ile ilgili yapılan bir çalışmaya göre Scrum ama Scrum modelinde temel prensipleri uzatarak veya kısaltarak uygulamayı gerektiren uyuşmazlık durumlarında ortaya çıkmaktadır. Çalışmada ayrıca Scrum master rolü içerisindeki değişken tanımlamaların, Scrum ama ortamının doğmasına yol açtığı üzerinde durulmaktadır (Attanasio, 2015).

2.8. Scrum ile İlgili Çalışmalar

Çevik yazılım geliştirme yöntemleri arasında yer alan Scrum yazılım geliştirme yönteminin organizasyonlarda kullanılmaya başlanması ile kazanılan faydaları, zararları, tecrübeleri, ve bu yöntemle gerçekleştirilen projelerin sonuçlarını konu alan bir çok farklı çalışma bulunmaktadır. Bu bölümde, Scrum yazılım geliştirme yöntemi ile ilgili alan yazında yer alan farklı çalışmalar ve sonuçları özetlenmektedir. Bu çalışmalarda Scrum'ın organizasyonlarda nasıl kullanılacağına ve Scrum pratiklerinin nasıl uygulanacağına yönelik bir çok tavsiye bulunmaktadır. Ayrıca geleneksel yazılım geliştirme sürecinden çevik yazılım sürecine geçecek organizasyonlar için Scrum'ın engel teşkil edebilecek yönlerine değinilmektedir.

Lene ve Jane Pries-Heje'nin (Pries-Heje, J. ve Pries-Heje, L., 2011) 2011 yılında, çalışanlarının %80'inin Danimarka'da %20'sinin ise Hindistan'da bulunduğu bir bankada gerçekleştirdiği araştırmada, Scrum kullanan dağıtık yazılım geliştirme projelerinde "Scrum neden çalışır?" sorusunun cevabı aranmaktadır. Makale sonuçlarında basit bir yapı olan günlük Scrum toplantısı ve beyaz tahta kullanımının Scrum takımını destekleyen bir unsur olduğu ve Scrum yazılım geliştirme yönteminin 4 yönüyle koordinasyonu sağladığı belirtilmektedir. Bunlar ürün iş listesi, Sprint iş listesi, günlük Scrum toplantısı ve beyaz tahta unsurlarıdır. Özellikle günlük Scrum toplantılarının takım içerisinde koordinasyonun ne zaman gerekli olduğunu anlamaya yarar sağladığı belirtilmektedir.

Pshigoda ve Smits'in (Pshigoda ve Smits, 2007) 2007 yılında yaptığı, dağıtık yazılım geliştirme organizasyonlarında Scrum uygulamalarına geçişin anlatıldığı makale sonuçlarına göre, Scrum uygulamasına başlanan projenin ilk 18 ayında alınan sonuçlar ümit verici olarak değerlendirilmektedir. Proje sonunda ürün sahibi

tarafından ürün yol haritası geliştirilerek ve ürün iş listesi tanımlanarak, çalışan yazılım teslim edilmiştir. Makalede projeye engel teşkil eden unsur olarak, Scrum metotlarının çalışanlara ve kaynaklara uygun olmaması gösterilmiştir. Scrum uygulamasına geçişteki geribildirimlerin pozitif olduğu ve Scrum'ın çalışanların planlama yeteneğini arttırdığı eklenmiştir.

Paasivaara, Durasiewicz ve Lassenius'un (Durasiewicz ve diğ., 2009) 2009 yılında yaptığı, 2 küçük ve 1 orta ölçekli organizasyonlarda yapılan dağıtık Scrum projelerinde kullanılan Scrum pratikleri (günlük scrum toplantısı, sprint iş listesi vs.) konulu makaleye göre, eğer Scrum uygulamasına geçecek takım daha önceki projelerde Scrum yazılım geliştirme yöntemini kullanmadı ise, öğrenmek için sadece Scrum materyalleri yeterli olmamaktadır. Konu ile ilgili Scrum takımı içerisinde görev alan personellere eğitim verilmesi gerekmektedir.

Scrum modelinin tekrarlı, artan ve deneyimsel bir süreç olduğu belirtilen ve 2016 yılında yapılan bir çalışmaya göre, Scrum modelinin organizasyonlara adaptasyonu sırasında karşılaşılan problemler, insanlar, süreç, proje ve organizasyonlar olarak dört grupta toplanmaktadır. Ayrıca çalışmada Scrum modelinin ürünlere değer katmaya odaklanarak, projelere açıklık ve şeffaflık sağladığı belirtilmektedir (Guerra-Garc ve diğ., 2016).

Geleneksel yazılım geliştirme yöntemini kullanan Türkiye'deki bir bankanın (Kır ve Özaydın, 2014), bu yöntemi kullanırken yaşadığı sorunlar ve çevik yazılım geliştirme yöntemlerinden Scrum'a geçişi sürecinde edinilen tecrübeleri konu alan tez çalışmasının sonuçlarına göre, geleneksel yazılım geliştirme sürecinden Scrum yazılım geliştirme yöntemine geçiş sırasında organizasyonlarda genel olarak yeni araçların ve metotların kullanımı ile personel eğitimi gibi konularda benzer sorunların yaşanmaktadır. Aynı çalışmada yer alan anket sonuçlarına göre Scrum yazılım geliştirme yöntemini kullanan personelin % 85'i bu yöntemi kullanmaktan memnun olduklarını belirtmektedir. Yukarıda örnekleri verilen Türkiye'den ve dünyadan Scrum yazılım geliştirme yönteminin organizasyonlarda kullanılmaya başlanmasına yönelik olarak yapılan çalışmalarda raporlanan Scrum ve ilgili pratikleri konusundaki olumlu ve olumsuz noktalar Tablo 2.1'de özetlenmektedir.

Tablo 2.1. Scrum ve Scrum pratikleri ile ilgili alınan geribildirimler

Olumlu geribildirimler	Olumsuz geribildirimler
Günlük Scrum toplantısı yapılması	Scrum metotlarının çalışanlara uygun olmaması
Beyaz tahta kullanımı	Scrum metotlarının kaynaklara uygun olmaması
Ürün ve Sprint iş listelerinin hazırlanması	Scrum materyalleri yeterli olmaması
Scrum'ın personelin planlama yeteneğini arttırması	Scrum için özel personel eğitimi gerekliliği

3. ARAŞTIRMA YÖNTEMİ

Tez çalışmasının bu bölümünde kullanılan araştırma yönteminin açıklaması yapılmaktadır. Araştırmanın stratejisi, tasarımı, araştırmanın uygulanma adımları, veri toplama araçları ve veri analizi sunulmaktadır. Çalışma kapsamlı bir yaklaşımla sorulan sorular ile tanımlayıcı bir olay inceleme şeklinde gerçekleştirilmiştir. Çalışma içerisinde uygulanan anket ve görüşme veri toplama araçları ile katılımcılara ve organizasyonlarına ait demografik özellikler raporlanmaktadır.

3.1. Araştırma Stratejisi

Bu çalışmada olay inceleme araştırma metodolojisi ile iki farklı yazılım organizasyonunda görev alan Scrum takımlarının profesyonelleri ile görüşmeler sağlanmıştır. Olay incelemesinde, yazılım organizasyonlarında çevik yaklaşımların ve özellikle Scrum yazılım geliştirme yönteminin uygulanma durumları ve geleneksel yazılım geliştirme yöntemlerinden çevik yazılım geliştirme yöntemlerine geçişte takım çalışanlarının yaşadıkları tecrübeler detaylı bir şekilde incelenmiştir. Bu olay inceleme görüşmelerine ek olarak daha geniş kapsamlı bir anket uygulanarak Türkiye'de yer alan organizasyonlarda çevik yazılım geliştirme modellerinin uygulanması ile ilgili durum tespiti de yapılmaya çalışılmış ve literatürdeki çevik yazılım geliştirme yöntemlerini inceleyen diğer çalışmalar ile desteklenerek uygulamadaki durum açıklanmaya çalışılmıştır. Görüşme ve anket sonuçları değerlendirildiğinde, Türkiye'de yer alan organizasyonlarda Scrum yazılım geliştirme modelinin farklı uygulanma şekilleri ile kullanıldığı izlenmiştir. Bu çalışmada amaç, Scrum yazılım geliştirme yönteminin organizasyonlarda uygulanan değişik özellikleri ile görüşülen profesyoneller tarafından önerilen özelliklerini tek bir model üzerinde birleştirerek, organizasyonlarda uygulanabilecek Harmanlanmış Scrum modelini ortaya çıkarmaktır.

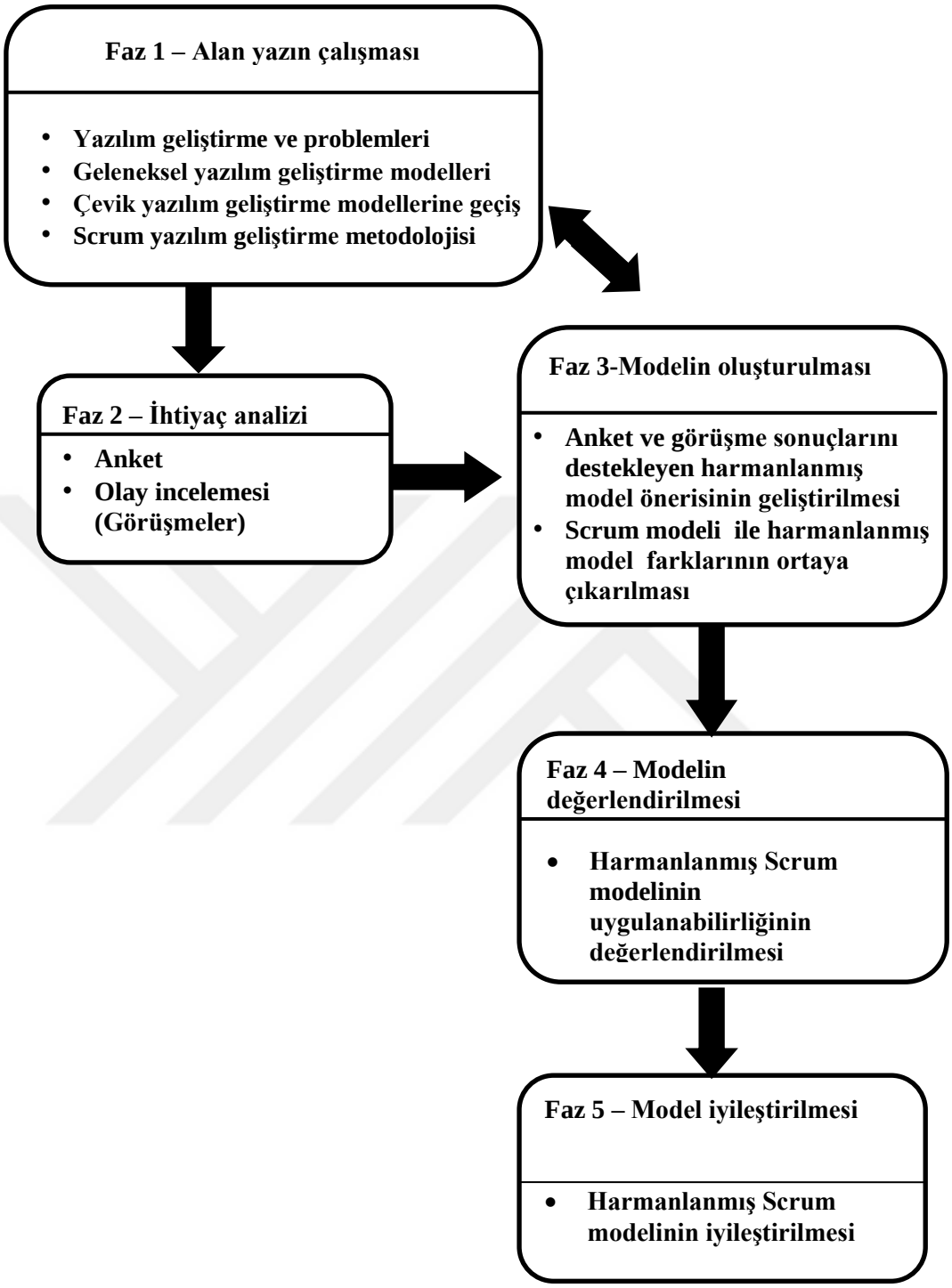
Olay inceleme araştırma metodolojisi, konu alınan olayın şimdiki durumunu ve çevreyle olan etkileşimini ilişkisel özellikleri ile birlikte değerlendirerek detaylı

şekilde incelenmesi olarak tanımlanmaktadır. Bir kişinin, bir yöntemin veya bir dokümanın ayrıntılı olarak incelenmesidir (Kazak, 2001). Neden ve nasıl soruları ile irdelenen olay incelemesinde, çalışmayı hazırlayan araştırmacının inceleme yaptığı olaylar üzerindeki kontrolü çok azdır (Yin, 2002). Bu çalışmada olay incelemesi çevik yazılım geliştirme modellerinden Scrum yazılım geliştirme yöntemini uygulayan organizasyonlarda ayrıntılı incelenmesi şeklinde yapılmıştır. Olay incelemesi içerisinde cevap aranan araştırma soruları aşağıdaki gibidir.

- Profesyonellerin Scrum ve çevik yazılım geliştirme ile ilgili deneyimleri ve görüşleri nelerdir?
- Geleneksel yazılım geliştirme süreçlerinden Scrum sürecine geçişte yaşanan deneyim ve zorluklar nelerdir?
- Bu zorlukları aşmak için ne gibi stratejiler geliştirilmiştir?
- Organizasyonlarda Scrum yazılım geliştirme metodolojisini kullanırken hangi kaynaklar ve metotlar kullanılmaktadır?
- Pratikteki uygulamalar Scrum yazılım geliştirme metodolojisine nasıl adapte edilebilir?

3.2. Araştırma Tasarımı / Araştırmanın Uygulanması

Araştırmanın gerçekleştirilme adımlarının grafik yardımı ile gösterimi Şekil 3.1'de izlenmektedir. Çalışma 5 fazdan oluşmaktadır. İlk faz olan alan yazın çalışmasında, yazılım geliştirmede geleneksel ve çevik yazılım geliştirme modelleri ile Scrum yazılım geliştirme yöntemi incelenerek sorunlar analiz edilmiş, ikinci fazda ise anket ve görüşme veri toplama araçları ile veriler toplanarak elde edilen bulguların raporlaması sağlanmıştır. Bu bulgular göz önüne alınarak hazırlanan Harmanlanmış Scrum modeline ilişkin detaylar ve Scrum modeli ile Harmanlanmış model farkları ise faz üç içerisinde yer almaktadır. Harmanlanmış modelin uygulanabilirliğinin değerlendirilmesi dördüncü fazda gerçekleştirilmiş ve sonuçları raporlanmıştır. Son faz olan beşinci fazda ise modelin iyileştirmesi ile ilgili çalışmalar gerçekleştirilmiştir.



Şekil 3.1. Araştırmanın gerçekleştirilme adımları

3.3. Veri Toplama Araçları

Olay incelemesi araştırma yönteminde veri toplama araçları aracılığı ile veri toplanırken birden çok veri kaynağı kullanılması yaygın olarak tercih edilmektedir (Yin,1984; Şimşek ve Yıldırım, 2013). Nitel ve nicel yöntemler olarak belirtilen veri toplama yöntemlerinden, nicel araştırma yöntemleri ölçülebilir, gözlenebilir ve

sayılarla gösterilebilir ifadelerden oluşmakta ve neden-sonuç ilişkilerini tanımlamaktadır (Glesne ve Peshkin, 1992). Nitel araştırma yöntemleri ise neden-sonuç ilişkisinden uzak olarak olayları yorumlayıcı bir yaklaşım ile incelemektedir (Husen, 1994; Yıldırım ve Şimşek, 2013). Diğer bir deyişle, nicel araştırma yöntemlerinde neden–sonuç ilişkisini açıklayan veriler, nitel araştırma yöntemlerinde öznel ve yoruma dayalıdır (Gall ve diğ., 1996).

Araştırma çalışmaları yapılırken mevcut soruların birden fazla yöntem ile yanıtlanması çeşitleme olarak tanımlanmaktadır (Denzin ve Lincoln 1994). Çeşitleme ayrıca nitel ve nicel araştırma yöntemlerinin birlikte kullanılması olarak belirtilmektedir. Araştırma soruları yanıtlanırken farklı araştırma yöntemlerinin kullanılması ile aynı sonuçların elde edilmesi güçtür, ancak nitel ve nicel yöntemlerden elde edilen veriler karşılaştırılarak çalışmanın güvenilirliği ölçülebilmektedir (Patton, 1990).

Bu tez çalışmasında, Türkiye'de yazılım organizasyonlarında görev alan profesyonellerin çevik yaklaşımlarla ilgili algılarının nasıl olduğu ve çevik yazılım geliştirme yöntemlerinin ne kadar kullanıldığı ile geleneksel yazılım geliştirme yöntemlerinden çevik yazılım geliştirme yöntemlerine geçişte organizasyonların yaşadıkları tecrübeleri belirlemek amacı ile incelemeler yapılmıştır. Nicel veriler anket veri toplama aracı ile toplanırken, nitel veriler ise görüşme veri toplama aracı ile elde edilmiştir.

Çalışma içerisinde kullanılan görüşme veri toplama aracı nitel araştırmalarda verilerin toplanması için kullanılan, en az iki kişi arasında sözlü olarak sürdürülen sözel bir iletişim tekniğidir; (Özgüven, 1980; Cohen ve diğ., 2011; Coll ve Chapman, 2000; Şimşek ve Yıldırım, 2013; Toluk Uçar ve Akdoğan, 2009). Görüşme yönteminde, önceden hazırlanmış bir grup soru yardımı ile karşıdaki kişinin duygu ve düşüncelerini tespit edilmeye çalışılmakta ve kişi deneyimlerini paylaşmaya yönlendirilmektedir (Seidman, 1991; Türnüklü, 2000).

Görüşme tekniği içerisindeki kuralların katılığına göre, yapılandırılmamış görüşme, yapılandırılmış görüşme ve yarı yapılandırılmış görüşme olarak üçe ayrılmaktadır (Türnüklü, 2000; Karasar, 2005).Yarı yapılandırılmış görüşme tekniğinde araştırmacı sormayı planladığı soruları hazırlamıştır, ancak görüşmenin akışı farklı sorularla

değiştirebilir veya kişi yanıtlarını detaylandırmaya yönlendirilebilir (Türnüklü, 2000; Şimşek ve Yıldırım, 2013). Yapılan görüşmeler yarı yapılandırılmış görüşme tekniği ile belirli bir soru seti hazırlanarak, biri büyük (banka) biri orta ölçekli (yazılım firması) 2 organizasyonda, iki Scrum Master ve bir Takım Üyesi olmak üzere, Scrum takımları içerisinde görev alan toplam 3 profesyonel ile gerçekleştirilmiştir.

Görüşmeler sırasında bazı soruların yanıtları, başka sorular içerisinde yanıtlanmış veya görüşme içerisindeki akış organizasyonlar içerisinde görüşülen kişilerin görev veya yetkinliklerine göre değişiklik gösterebilmiştir. Görüşme zamanı profesyonellerin çalışma durumları göz önünde bulundurularak belirlenmiş, görüşme ortamı ise farklı organizasyonlarda yer alan profesyonellerin çalışma ortamları görülecek şekilde planlanmıştır. Görüşmeler profesyonellerden alınan izin doğrultusunda ses kayıt cihazı ile kayıt edilmiştir. Görüşme soruları Ek-B bölümünde yer almaktadır.

Scrum yazılım geliştirme yönteminin büyük ve orta ölçekli organizasyonlarda nasıl uygulandıklarını anlamaya yönelik hazırlanan 52 madde olarak hazırlanan görüşme soruları altı bölümde gruplanmıştır. Birinci bölüm profesyoneller, yer aldıkları organizasyonlar ve projeleri ile ilgili demografik bilgileri almaya yönelik olarak hazırlanmıştır. İkinci bölümde çevik yazılım geliştirme kavramı ve uygulaması odaklı sorular yöneltilmiştir. Üçüncü bölümde ise daha önce geleneksel yazılım geliştirme yöntemini kullanan profesyonellerin organizasyonlarını Scrum yazılım geliştirme yöntemine adapte ederken karşılaştıkları durumlar ile kullandıkları Scrum rolleri ve pratiklerine yönelik sorular yer almaktadır. Görüşmenin dördüncü bölümünde katılımcılara Scrum yazılım geliştirme yöntemini uygularken kullandıkları araçlar, bu araçlarla ilgili memnuniyetleri ve araçlara eklemek veya çıkarmak istedikleri özellikler hakkında sorular sorulmuştur. Beşinci bölümde, çevik yazılım geliştirme projelerinin yönetimi ve organizasyonlarındaki başarı durumları ile ilgili sorular yöneltilmiş; altıncı bölümde ise çevik yazılım geliştirme ile ilgili algıları belirlenmeye çalışılmıştır.

Çalışmada kullanılan nicel veri toplama aracı olan anket içerisinde, Türkiye'deki yazılım organizasyonlarında çevik ve geleneksel yazılım geliştirme yöntemlerinin kullanım durumu ve çevik yaklaşımlarla ilgili profesyonellerin görüşlerini analiz

etmek için 39 maddeden oluşan sorular bulunmaktadır. Üç temel bölümden oluşan anketin, birinci bölümü katılımcılarla ilgili demografik bilgileri almaya yönelik hazırlanmıştır. Bu bölüm sonunda katılımcılara organizasyonlarında hangi yazılım geliştirme yöntemini kullandıkları sorularak, çevik yöntemleri seçenlerin ikinci bölümde yer alan çevik yazılım geliştirme odaklı soruları yanıtlamaları sağlanırken, geleneksel yöntemi seçenlerin ise üçüncü bölümde yer alan geleneksel yönteme yönelik soruları cevaplamaları sağlanmıştır. Bu sorular, geleneksel yazılım geliştirme yöntemlerini kullanan organizasyonlarda yer alan katılımcıların çevik yaklaşımlar konusundaki görüşlerini almaya yönelik olarak hazırlanmıştır. Geleneksel yöntem kavramı ile belirtilmek istenen, çevik yöntemler dışında kalan organizasyonların kullandığı yazılım geliştirme süreci modelleridir. Çevik yöntemlere yönelik hazırlanan 23 maddeden 19 tanesi Ambler'in (2008) çalışmasından Türkçeye çevrilerek adapte edilmiştir. Geleneksel yöntemlere yönelik hazırlanan sorulardan 6 maddeden üç tanesi Baytam ve Kalıpsız'ın (2011) çalışmasından ve 1 tanesi yine Ambler'in (2008) çalışmasından adapte edilmiştir. Bu kapsamda, hazırlanan sorular konu ile ilgili daha önce anket tasarlamış ve bu konu da eğitim veren kişiler tarafından gözden geçirmeleri için iletilmiş ve verdikleri geribildirimler doğrultusunda güncellemeler yapılmıştır. Böylece çalışmanın yüzeysel ve içerik geçerliliği sağlanmıştır (Black ve Champion, 1976). Anket soruları Ek-A'da yer almaktadır.

Hazırlanan anket, Surveey isimli bir çevrimiçi anket sağlama hizmeti kullanılarak katılımcıların erişimine açılmıştır. Katılımcılar çeşitli sosyal ağlar, e-mail grupları kullanılarak davet edilmişlerdir. Yazılım şirketleri, bankalar, telekomünikasyon kurumları ve bilişim sektöründe yer alan birçok farklı kuruma ulaşılarak, 74 yazılım geliştiricinin görüşleri alınmıştır.

Çalışma kapsamındaki dördüncü faz sırasında, Harmanlanmış Scrum modelinin gerçek dünyaya uyumunu ve profesyonellerin bu modele bakış açılarını belirlemek için, model içerisinde yer alan özellikler bir değerlendirme listesi haline getirilerek yalnızca Scrum yöntemini organizasyonlarında uyguladıklarını belirten profesyonellere sunulmuştur. Harmanlanmış model değerlendirme maddeleri Ek-D'de yer almaktadır.

74 adet olarak hazırlanan Harmanlanmış model değerlendirme maddeleri beş bölümde gruplanmıştır. Birinci bölüm profesyoneller ve görev aldıkları organizasyonlar ile ilgili demografik bilgileri almaya yönelik olarak hazırlanmıştır. İkinci bölümde Harmanlanmış Scrum modelinin rolleri ile ilgili, üçüncü bölümde ise modelin pratikleri ile ilgili sorular yer almaktadır. Değerlendirmenin dördüncü bölümünde katılımcılara Harmanlanmış Scrum modelini uygularken kullanılacak Scrum araçlarına yönelik sorular sorulmuştur. Beşinci bölümde ise, Harmanlanmış Scrum modelinin Sprint süreci ile ilgili sorular yöneltilerek profesyonellerin modele ile ilgili genel bakış açıları belirlenmeye çalışılmıştır. Katılımcılardan değerlendirme listesinde yer alan 6 aşama içerisindeki maddeler için "Fikrim yok", "Kesinlikle Katılmıyorum", "Katılmıyorum", "Kararsızım", "Katılıyorum", "Kesinlikle Katılıyorum" şeklinde görüş belirtmeleri istenerek önerilen Harmanlanmış Scrum modelinin uygulanabilirliğini değerlendirmeleri sağlanmıştır.

3.4. Veri Analizi

Çalışma kapsamında elde edilen verilerin analiz edilmesi için nitel ve nicel veri analizi yöntemlerinden faydalanılmıştır. Nitel veri analizi, gözlem ve görüşme gibi veri toplama yöntemleri ile elde edilen verilerin düzenlenerek, kategorilere ayrıldığı ve temalarının oluşturularak raporlandığı bir yöntemdir. Nitel araştırma yapan kişi, verilerin arasında saklanan bilgiyi ortaya çıkarmaktadır (Ozdemir, 2010). Nitel araştırma diğer bir deyişle, insanların olayları öznel bakış açıları ile nasıl tanımladıklarını araştırmaktadır (Dey, 1993; Storey, 2007). Nitel veri analizinde veri toplama yöntemleri ile elde edilen veriler içerisinde yer alan kelime veya cümleler ayıklanarak kodlanmaktadır. Bu kodlara kategoriler tanımlanmaktadır. Farklı kategorilerin isimlendirmeleri yapılarak, aralarında ilişki bulunan kategoriler aynı tema altında toplanmaktadır (Marshall ve Rossman, 1995; Ratcliff, 2008).

Görüşmelerde nitel veri analiz türleri arasında en sık kullanılan yöntemlerden biri olan içerik analizi adımları uygulanmıştır. Verileri analiz etmek için kullanılan üç yöntemden biri olan, içerik analizi yönteminde temel amaç, toplanan verileri açıklayıcı kavramlarla gösterebilmektir. İçerik analizinde benzer alanlar belirli kavramlar ile bir araya getirilerek verilerin kodlanması işlemi yapılır (Wolcott, 1994; Sandelowski, 2000; Silverman, 2001). İçerik analizinde konular araştırmacı

tarafından kategorilere ayrılmaktadır. Bu kategoriler oluşturulurken benzer araştırmalarda aynı sonuçların verileceği türden kategoriler geliştirmeye dikkat edilmelidir (Silverman, 2001).

Çalışma içerisinde uygulanan anket çalışması için kullanılan veri analizi, istatistiksel veri analizine dayalı nicel bir yöntemdir (Alain ve Keneth, 1993). Ankete dayalı tanımlayıcı bir inceleme şeklinde gerçekleştirilen çalışmada amaç, değişkenler arası ilişkilerin belirlenmesinden çok örneklemin ne kadarının belirli bir fikirde olduğu veya bazı olayların ne sıklıkla olduğunun belirlenmesidir (Oppenheim, 1996). Ancak sayısal veriler profesyonellerin deneyimlerini tanımlamak için yeterli olmayabilir. Bu yüzden nicel araştırmalar nitel araştırmaların sonuçları ile güçlendirilebilmektedir (Bryman, 1992).

Çalışma içerisinde, tablolar yardımı ile organizasyonlarında Scrum yazılım geliştirme yöntemini kullanan kişiler ile yapılan görüşmelerin sonuçları karşılaştırmalı olarak değerlendirilmektedir. Ayrıca, daha önce geleneksel yazılım geliştirme yöntemlerini kullanan organizasyonların, çevik yazılım geliştirmeye adapte olduktan sonra, Scrum yazılım geliştirme yöntemini nasıl uyguladıkları ve teorideki Scrum modeli ile organizasyonlarda uygulanan Scrum modelinin karşılaştırması yapılmaktadır.

Görüşme sonuçları, organizasyon bazında veya profesyonellerin görev yetkinlikleri göre kodlanarak tablolar yardımı ile gösterilmiştir. Ayrıca görüşülen profesyonellerin verdiği cevaplar kendi sözleri ile ilgili bölüm içerisinde belirtilmiştir. İçerik analizi adımları uygulanan görüşmeler ile ilgili raporlamalar içerisinde yer alan kodlamalar ise araştırmacı tarafından yapılmıştır. İhtiyaca göre geliştirilen kodlar Tablo 3.1'de yer alan "Kod Açıklama Tablosu" içerisinde izlenmektedir.

Görüşme sonuçları, banka organizasyonu için "O1" (Organizasyon 1) , yazılım firması için ise "O2" (Organizasyon 2) şeklinde kodlanarak, cevaplar profesyonellerden alınan geribildirimlere göre tablolar kullanılarak gösterilmiştir. Kodlama içerisinde yer alan "SM" ifadesi Scrum master, "TU" ifadesi ise takım üyesi görevlerini ifade etmektedir. İki organizasyonda da Scrum yazılım geliştirme yöntemini uygulayan takımlar bulunmaktadır. O1 içerisinde 1 adet Scrum takımı, O2 içerisinde ise 2 adet Scrum takımı bulunmaktadır.

Tablo 3.1. Kod açıklama tablosu

Kod	Açıklama
O1	Organizasyon 1
O2	Organizasyon 2
SM	Scrum Master
TU	Takım Üyesi
O1_SM	Organizasyon 1 Scrum Master
O1_TU	Organizasyon 1 Takım Üyesi
O2_SM	Organizasyon 2 Scrum Master

Görüşmeler esnasında profesyonellerden alınan yanıtlar ve yöneltilen soruların genel yapıları dikkate alınarak hazırlanan temalar ve bunlar içerisinde yer alan kategorilere ait kodlamalar Tablo 3.2'de izlenmektedir. Tabloda izlenen temalar, organizasyonlarda uygulanan çevik yazılım geliştirme kavramı ve uygulaması, Scrum yazılım geliştirme yönteminin genel yapısı, geleneksel yazılım geliştirme yönteminden Scrum yazılım geliştirme yöntemine geçiş ve organizasyonda kullanılan proje yönetim aracı şeklindedir. Kategorilere ait detaylar ise kategori numaraları ile birlikte gösterilmektedir. İçerik analizinin katılımcıların verdikleri yanıtlara dayalı olan tüm detayları Ek-C'de paylaşılmaktadır.

Tablo 3.2. Görüşmelerin kodlaması

Tema	Kategori No	Kategori tanımlaması
Organizasyonlarda uygulanan çevik yazılım geliştirme kavramı ve uygulaması	KN_1_1	Çevik yöntem ile ilgili eğitimli personel
	KN_1_2	Çevik yöntemin kullanım süresi
	KN_1_3	Çevik yöntemin kullanıldığı proje sayısı
	KN_1_4	Çevik yöntem kullanan proje takımının büyüklüğü
	KN_1_5	Çevik yöntemin proje yönetimine sağladığı avantaj ve dezavantajlar
Scrum yazılım geliştirme yönteminin genel yapısı	KN_2_1	Organizasyonlarda yer alan Scrum rolleri
	KN_2_2	Organizasyonlarda Scrum pratikleri kullanımı
	KN_2_3	Organizasyonlarda Scrum Toplantıları Süreleri
	KN_2_4	Organizasyonlarda Scrum kavramları kullanımı
Geleneksel yazılım geliştirme yönteminden Scrum yazılım geliştirme yöntemine geçiş	KN_3_1	Geleneksel yöntemlerden Scrum yöntemine geçiş süresi
	KN_3_2	Geleneksel yöntemlerden Scrum yöntemine geçişte zorlanılan durumlar
Organizasyonda kullanılan proje yönetim aracı	KN_4_1	Proje yönetim aracı kullanımı
	KN_4_2	Proje yönetim aracının genel özellikleri ve memnuniyet algısı
	KN_4_3	Proje yönetim aracına eklenmesi gereken özellik
	KN_4_4	Proje yönetim aracından çıkarılması gereken özellik

Çalışmanın dördüncü fazında uygulanan değerlendirme çalışması için, nicel bir yöntem olan istatistiksel veri analizi (Alain ve Keneth, 1993) kullanılmıştır. Yapılan değerlendirme ile Harmanlanmış Scrum takımı rol ve sorumlulukları, Harmanlanmış Scrum modeli toplantıları, araçları ve Sprint süreci ile ilgili değerlendirme maddeleri için profesyonellerden alınan görüşler analiz edilerek ilgili başlıklar altında raporlanmıştır.

3.5. Katılımcılar ve Organizasyonlar

3.5.1. Anket katılımcıları ve organizasyonlarına ait bilgiler

Çalışma kapsamında gerçekleştirilen ankete katılan 74 katılımcının eğitim durumlarına bakıldığında 31'inin üniversite, 24'ünün yüksek lisans, 12'sinin ise doktora derecesinde olduğu, 7'sinin ise ön lisans mezunu olduğu izlenmektedir. Bu veriler ile anket katılımcılarının büyük çoğunluğunun üniversite ve üzeri düzeyde eğitim aldığı görülmektedir. Görev alınan pozisyon olarak ilk üç sırada yazılımcı, proje lideri ve takım üyesi yer almaktadır. Diğer taraftan katılımcıların en az görev aldıkları pozisyonlar ise, grafikte 5 kişinin altında katılımcı sayısı ile izlediğimiz ürün sahibi, testçi ve operasyonel destekte görev alan katılımcılar oluşturmaktadır. Ankette yer alan ve 10 kişiden fazla katılımcının tercih ettiği "diğer" alanı ise; genel müdür yardımcısı, sistem mühendisi, sistem yöneticisi gibi pozisyonları içermektedir. Bu soru çoklu seçimli bir soru olduğu için, katılımcıların 4'ü 2 farklı pozisyonda, 4'ü 3 farklı pozisyonda ve 3'ü ise 4 farklı pozisyonda görev almaktadır. 3 kişi ise 4 farklı pozisyonda görev almaktadır. 63 katılımcı ise sadece tek pozisyonda görev aldıklarını bildirmiştir.

Katılımcıların görev aldıkları firmaların hizmet verdiği sektörleri sırasıyla Finans-Banka, Telekomünikasyon ve Bilgi Teknolojileri sektörleri oluşturmaktadır. 5 katılımcı Devlet sektöründe hizmet verirken, Sağlık ve Sigorta sektörleri ise en az katılımcının yer aldığı sektörler olarak izlenmektedir. Ankette yer alan diğer alanı ise; robotik teknolojiler, hayvancılık, savunma sanayi, otomasyon sistemleri, üniversite ve sigorta sektörlerinin tümünü içermektedir. Katılımcıların cinsiyet, eğitim durumları, çalıştıkları firmalarda görev aldıkları pozisyonlar ve firmaların sektörlerine ait bilgiler Tablo 3.3'te gösterilmektedir.

Tablo 3.3. Anket katılımcıları demografik bilgileri

	Kişi Sayısı
Eğitim Durumu	
Üniversite	31
Yüksek Lisans	24
Doktora	12
Ön Lisans	7
Pozisyon	
Yazılımcı	27
Proje Lideri	14
Takım Üyesi	12
Diğer	11
İş Analisti	8
IT Müdürü	6
Geliştirici	5
Operasyonel Destek	3
Ürün Sahibi	3
Testçi	3
Sektör	
Finans – Banka	28
Telekominikasyon	18
Bilgi Teknolojileri	14
Devlet	5
Diğer	7
Sigorta	1
Sağlık	1

3.5.2. Olay incelemesi kapsamında görüşülen katılımcılar ve organizasyonlarına ait bilgiler

Çalışma kapsamında O1 ve O2 şeklinde kodlanan iki farklı organizasyon incelenmiştir. Bu organizasyonlarda Scrum master olarak görev alan O1_SM, O2_SM ve takım üyesi olarak göre alan O1_TU profesyonelleri ile yapılan görüşmelerde organizasyonlarında bulunan personel sayısı, Scrum kullanım süresi, mevcut Scrum rolleri ve hizmet verdikleri sektör ile ilgili bilgiler alınmıştır.

O1'de personel sayısı 6600-7000 arasında iken O2'de 200-250 arasında değişmektedir. O1 bir banka, O2 ise bir yazılım firmasıdır. Banka içerisinde yer alan personel sayısı tüm organizasyon içerisinde çalışan personelleri belirtmektedir. O2 ise sadece yazılım geliştiren bir organizasyon olduğundan dolayı çalışan personelin büyük çoğunluğu yazılım geliştirme safhalarında yer almaktadır. O1 içerisinde 3 yıldan fazla bir süredir Scrum kullanılırken, O2'de Scrum kullanım süresi 4 yılı aşkın

olarak izlenmektedir. Organizasyonda yer alan Scrum rolleri her ikisinde de Ürün sahibi, içerisinde analist, testçi ve yazılım geliştiricilerin bulunduğu Takım Üyeleri ile Scrum Master olarak belirtilmektedir. O1 bankacılık sektöründe, O2 ise yazılım sektöründe hizmet vermektedir. Organizasyonlara ait bilgiler Tablo 3.4'te görülmektedir.

Tablo 3.4. Organizasyonların özellikleri ile ilgili demografik tablo

Kod	Organizasyonun Personel Sayısı	Organizasyonda Scrum kullanım süresi	Organizasyonda yer alan Scrum Rolleri	Organizasyonun hizmet verdiği sektör
O1	6600 – 7000	3-4 yıl	Ürün Sahibi, Takım Üyeleri (Analist, Testçi, Yazılım Geliştirici) ScrumMaster	Banka
O2	200 -250	4-5 yıl	Ürün Sahibi, Takım Üyeleri (Analist, Testçi, Yazılım Geliştirici) ScrumMaster	Yazılım Şirketi

İncelenen organizasyonlarda görüşülen SM ve TU'lerin yazılım tecrübeleri, eğitim durumları, görev aldıkları birim, şimdiye kadar görev aldıkları toplam Scrum proje sayısı ve Scrum takımı içerisindeki görevleri Tablo 3.5'te karşılaştırmalı olarak gösterilmiştir. Görüşülen profesyonellerin üçü de yüksek lisans mezunudur ve organizasyonlarda farklı birimlerde görev almaktadır. O1_SM'nin yazılım tecrübesi 5 yılı, O1_TU'nun 6 yılı, O2_SM'nin ise 9 yılı aşkın olarak izlenmektedir. O1_SM daha önce 4 farklı projede, O1_TU ise 7'den fazla Scrum projesinde görev alırken, O2_SM ise göreve başladığından beri sürekli yeni sürümü çıkarılan aynı projede yer almaktadır.

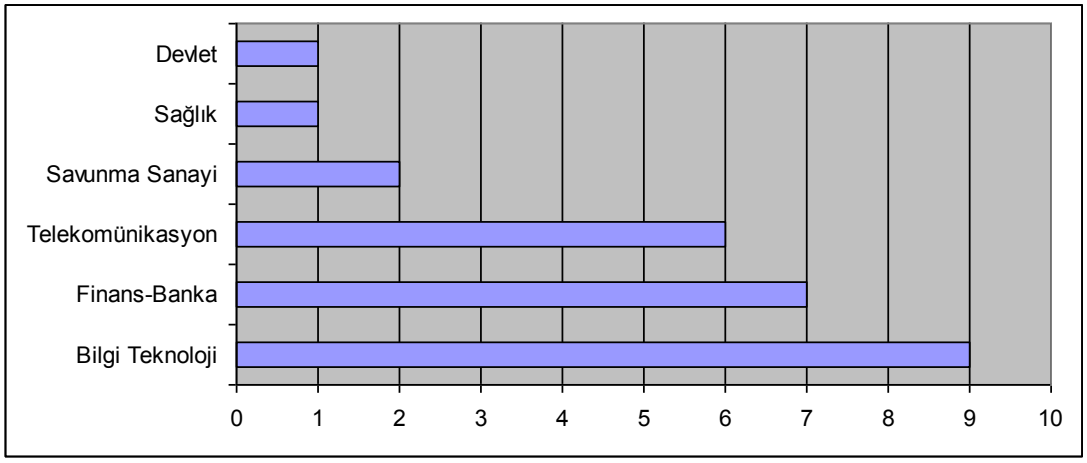
Tablo 3.5. Profesyonellerin eğitim ve tecrübeleri ile ilgili demografik tablo

Kod	Yazılım Tecrübesi	Eğitim	Görev Aldığı Birim	Görev Aldığı Scrum Proje Sayısı	Scrum Takımı İçerisindeki Görevi
O1_SM	5 yıldan fazla	Endüstri Mühendisliği -Yüksek Lisans Derecesi	Proje Yönetim Birimi	4	Scrum Master
O1_TU	6 yıldan fazla	Bilgisayar Mühendisliği - Yüksek Lisans Derecesi	Bilgi Teknolojileri ve Hizmetleri	7-8	Sistem Analist-Takım Üyesi
O2_SM	9 yıl	Bilgisayar Mühendisliği - Yüksek Lisans Derecesi	Ürün Geliştirme Birimi	Aynı proje içerisinde sürekli yeni sürüm çıkarılmaktadır.	Scrum Master

3.5.3. Harmanlanmış Scrum modeli değerlendirme katılımcıları ve organizasyonlarına ait bilgiler

Çalışma kapsamında gerçekleştirilen değerlendirmeye katılan 26 katılımcının demografik özellikleri izlendiğinde, 10 tanesinin kadınlardan, 16 tanesinin ise erkeklerden oluştuğu gözlemlenmektedir. Yaşları 24 ila 45 arasında değişiklik gösteren katılımcıların eğitim durumlarına bakıldığında 11'inin üniversite, 13'ünü yüksek lisans, 1'inin doktora, 1'inin ise yüksek lisans derecesinde olduğu izlenmektedir. Görev alınan pozisyon olarak ilk sırada proje lideri pozisyonu yer alırken; diğer katılımcılar arasında Scrum takım üyeleri, ürün sahibi, iş analisti, yazılımcı, BT müdürü ve geliştirici pozisyonları da bulunmaktadır. Katılımcıların görev aldıkları birimler içerisinde ilk sırayı sistem geliştirme ve yazılım geliştirme birimleri alırken, ürün geliştirme birimi de onları takip etmektedir. Değerlendirmeye ayrıca bilgi teknolojileri, süreç yönetimi, Ar-Ge ve ürün yönetimi gibi birimlerden de katılımcılar olmuştur.

Katılımcıların görev aldıkları firmaların hizmet verdiği sektörler sırasıyla Bilgi Teknolojileri, Finans-Banka ve Telekomünikasyon olarak izlenirken; Devlet, Sağlık ve Savunma Sanayi sektörlerinden katılımcılarda görev almaktadır. Katılımcıların çalıştıkları firmaların sektörlerine ait bilgiler Şekil 3.2'de gösterilmektedir.



Şekil 3.2. Değerlendirme katılımcıları organizasyonlarının hizmet sektörleri

Katılımcıların yazılım alanındaki iş tecrübesinin dağılımları Tablo 6.1'de yıl bazında olarak gösterilmektedir. Ortalama değer yaklaşık 10 yıldır. Katılımcıların 12'sinin 11 yıl ve üzeri iş deneyimlerinin olduğu izlenmektedir. Bu da değerlendirme katılımcılarının yazılım alanındaki tecrübelerinin fazlalığını göstermektedir. Tablo 3.6'da ayrıca katılımcıların BT sektöründeki toplam iş tecrübeleri ve bulundukları kurumdaki iş tecrübeleri de gösterilmektedir. Katılımcıların BT sektöründeki iş tecrübelerinin, genel iş tecrübeleri ile büyük oranda paralellik gösterdiği izlenmektedir.

Tablo 3.6. Değerlendirme katılımcıları yıl bazındaki tecrübeleri

	Ort.	<=5	6-10	11-15	16-24	>25
Katılımcıların iş deneyimleri	10	5	12	5	4	0
Katılımcıların BT sektöründeki toplam iş tecrübeleri	10	7	11	4	4	0
Katılımcıların bulundukları kurumdaki iş tecrübesi	5	21	2	1	2	0

Çalışma kapsamında gerçekleştirilen değerlendirmeye katılan katılımcıların organizasyonlarında bulunan personel sayısı, Scrum kullanımları, mevcut Scrum rolleri ve hizmet verdikleri sektör ile ilgili bilgiler alınmıştır. Katılımcıların büyük çoğunluğu (16) 5 yılın altında bir süredir organizasyonlarında Scrum yazılım geliştirme yöntemini kullandıklarını belirtirken, bir kısmı ise 5 yıl ve üzerinde bir süredir Scrum yöntemini uyguladıklarını ilettiler. Mevcut durumda organizasyonlarında Scrum sürecini kullanan katılımcıların sayısı 16 olarak izlenirken, diğer katılımcılar ise daha önceki tecrübelerinde Scrum sürecini

uyguladıklarını belirtmektedirler. Katılımcıların büyük çoğunluğu en fazla 2 Scrum projesinde görev aldıklarını belirtirken, bir kısmı ise 10 ve üzeri Scrum projesinde görev aldığını belirtmektedir.

Katılımcıların çalıştıkları organizasyonların toplam personel sayısına bakıldığında, 500'den küçük olanlar 13 kişi, 500 ila 999 arasında olanlar 5, 1000 ila 4999 arasında olanlar 6, 5000 ve üzeri olanlar ise 2 katılımcı olarak izlenmektedir. Organizasyonların toplam personel sayıları ile ilgili bilgiler Tablo 3.7'de görülmektedir.

Tablo 3.7. Organizasyonların toplam personel sayıları ile ilgili tablo

	Ort.	<=499	500-999	1000-4999	>5000
Katılımcıların organizasyonlarının toplam personel sayısı	4207	13	5	6	2

4. ANKET VE GÖRÜŞME YÖNTEMİ İLE ÇEVİK SÜREÇ İNCELEMENİN BULGULARI

Çalışmanın bu bölümünde, Türkiye'deki yazılım organizasyonlarında çevik ve geleneksel yaklaşımların ne kadar kullanıldığı ve yazılım geliştiricilerin bu yaklaşımlarla ilgili algılarının nasıl olduğunu belirlemeye yönelik hazırlanan ankete ait sonuçlar raporlanmaktadır. Ayrıca, Scrum yazılım geliştirme yönteminin büyük ve orta ölçekli organizasyonlarda nasıl uygulandıklarını anlamaya yönelik olarak örnek olay incelemesi kapsamında yapılan görüşmelere ait bulgular da yine bu bölümde paylaşılmaktadır.

4.1. Anket Bulguları

Çalışmanın bu bölümünde anket çalışmasından elde edilen bulgular yazılım geliştirme yaklaşımları, katılımcıların iş tecrübeleri, çevik yazılım geliştirme yaklaşımlarının uygulanması ve geleneksel yazılım geliştirme yaklaşımlarının uygulanması başlıkları altında raporlanmaktadır. Anket bulguları ilgili başlıklar altında tablolar yardımı ile gösterilmektedir.

4.1.1. Yazılım geliştirme yaklaşımları

Katılımcıların organizasyonlarında hangi yazılım geliştirme yöntemini kullandıkları sorusuna alınan yanıtlara göre, 40'ı çevik yazılım geliştirme, 34'ü ise geleneksel yazılım geliştirme yöntemlerini kullanmaktadır. Tablo 4.1'de görüldüğü üzere, en yüksek tercih çevik yazılım geliştirme yöntemleri içerisinden Scrum üzerine yoğunlaşmaktadır. Toplam 38 katılımcı Scrum yazılım geliştirme modelini kullanırken, 18 katılımcı geleneksel yazılım geliştirme yöntemlerinden Çağlayan modelini kullandıklarını belirtmişlerdir.

Tablo 4.1. Organizasyonlarda kullanılan yazılım süreç modelleri

Yazılım Süreç Modeli	Sayısı
Scrum	38
Çağlayan	18
XP	9
Artırımlı	9
Spiral	7
Evrimsel	6
V Model	5
Diğer	4
RUP	1

4.1.2. Anket katılımcıları iş tecrübeleri

Anket katılımcıların yazılım alanındaki iş tecrübesinin dağılımı ise Tablo 4.2'de yıl bazında gösterilmektedir. Ortalama değer yaklaşık 8 yıldır. Katılımcıların 25 ve 30'unun iş deneyimi 5 ila 10 yıl ve altındadır. Bu da katılımcıların çoğunluğunu genç bir kitlenin oluşturduğunu bize göstermektedir. Tablo 4.2'de ayrıca katılımcıların BT sektöründeki toplam iş tecrübeleri de gösterilmektedir. Katılımcıların bu sektördeki iş tecrübesi genel iş tecrübeleri ile büyük oranda paralellik göstermektedir. Katılımcılara şu an görev yaptıkları kurumda ne kadar süredir çalıştıkları sorulduğunda ise, birçoğunun (66) bulundukları firmada 10 yıla kadar deneyim sahibi oldukları görülmektedir. Anket aracılığı ile elde edilen verilere göre, yazılım alanındaki iş tecrübelerinin ortalama 8 yıl olduğu belirtilen katılımcıların, eğitim seviyelerinin ise 67 kişi ile üniversite ve üzeri olduğu belirtilmektedir.

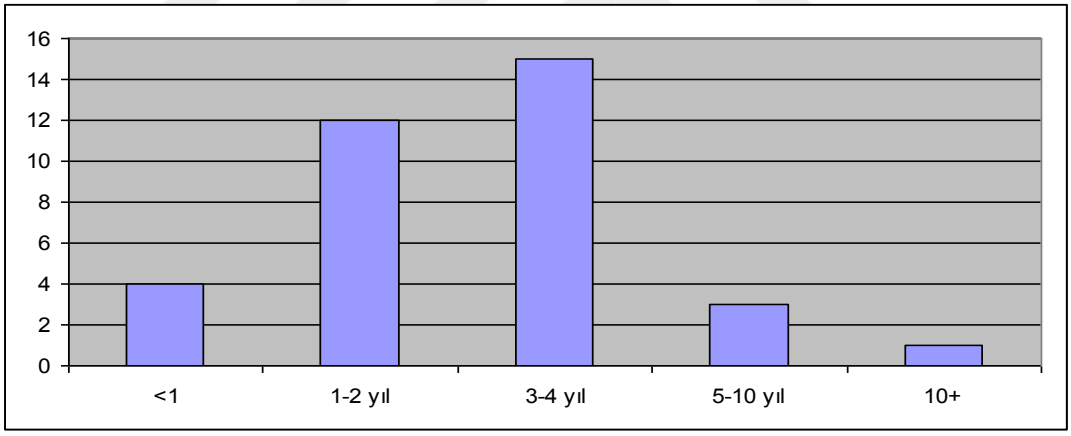
Tablo 4.2. Katılımcıların yıl bazındaki tecrübeleri

	Ort.	<=5	6-10	11-15	16-24	>25
Katılımcıların iş deneyimleri	8,2	25	30	15	3	1
Katılımcıların BT sektöründeki toplam iş tecrübeleri	8,2	27	26	17	3	1
Katılımcıların bulundukları kurumdaki iş tecrübesi	4,7	52	14	7	0	1

4.1.3. Çevik yazılım geliştirme yaklaşımlarının uygulanması

Anket kapsamında organizasyonlarında çevik yöntemleri kullandıklarını belirten katılımcılara, ankette çevik yazılım geliştirme yaklaşımlarının uygulanması ve geliştiricilerin bu yaklaşımlara yönelik algılarının belirlenmesine yönelik sorular yöneltilmiştir. Organizasyonlarında ne kadar süredir çevik yöntemlerin uygulandığı, uygulanan proje sayıları ve türleri, hangi çevik yaklaşımın daha çok uygulandığı, uygulanan yöntemlerde oluşturulan proje takımları ve kullandıkları destekleyici yazılımlar sorgulanmıştır.

Anket katılımcılarının organizasyonlarında çevik yaklaşımlar ile proje yürütülme süresine yönelik bulguları Şekil 4.1'de gösterilmektedir. Katılımcıların % 43'ü 3-4 yıldır, % 34'ü ise 1-2 yıldır çevik yaklaşım projelerinde görev aldıklarını belirtmektedir. Grafikte izlenebildiği üzere çevik yaklaşımların geçmişi birçok şirket için 3-4 yılı geçmemektedir.



Şekil 4.1. Organizasyonlarda çevik yöntemler ile proje yürütülme süresi (yıl bazlı)

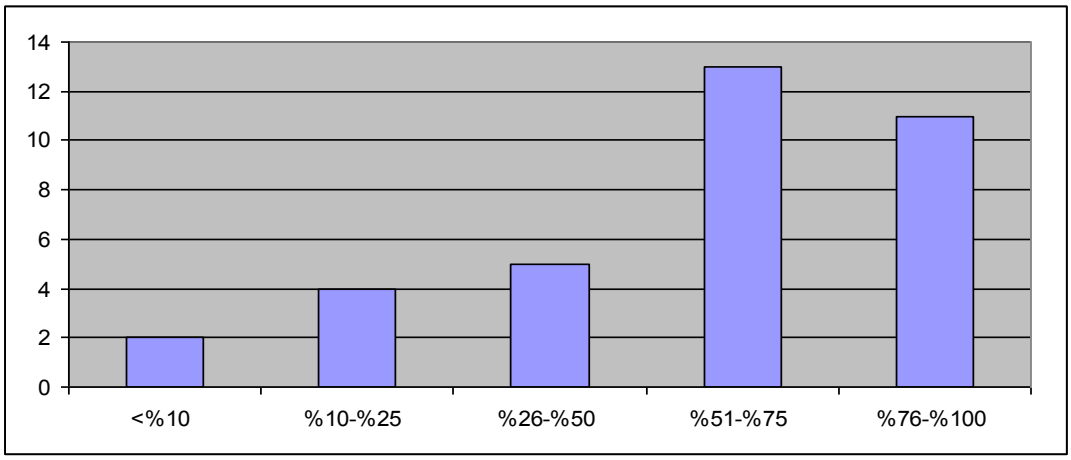
Çevik yaklaşım projelerinde çoğunlukla Scrum metodu ile çalışan katılımcıların görev aldıkları en büyük takım %23 ile 6 ila 10 kişilik takımlar ve yine % 23 ile 51 kişi üzerinde olan takımlar olarak izlenmektedir. Anket katılımcılarının çevik yaklaşımlarda 51 kişi ve üzeri takımlar ile çalışmasının sebebi olarak, birçoğunun 1000 ve üzeri personel çalıştıran finans-banka ve telekomünikasyon gibi sektörlerde görev aldıkları gösterilebilmektedir. Çevik projelerde genellikle tercih edilen takım büyüklüğü, % 51 ile 6 - 10 kişi arasındadır.

Çevik yöntemlerin kullanıldığı projelere veriler ile ilgili detaylar Tablo 4.3'te izlenmektedir. Bu verilere göre, çevik yaklaşımların genellikle en fazla 10 kişilik takımlar için tercih edildiğini söylenebilir.

Tablo 4.3. Çevik yöntem kullanılan proje bilgileri

	1-5	6-10	11-20	21-50	51+
	%	%	%	%	%
Kurumunuzda çevik yaklaşımları kullandığınız proje sayısı	56	23	6	9	6
Kurumunuzda çevik yaklaşımlar ile çalışan en büyük projede çalışan sayısı	14	23	20	20	23
Kurumunuzda çevik projelerde ortalama proje takımlarındaki kişi sayısı	34	51	9	3	3

Çevik yöntemleri kullandıklarını belirten katılımcılardan elde edilen anket verilerine göre, organizasyonlar içerisinde yazılım geliştirme sürecini destekleme için kullanılan en yaygın araçlar olarak VersionOne ve TFS (Team Foundation Server - Takım ortak bilgi yönetim sunucusu) öne çıkmıştır. Anket katılımcılarına çevik yazılım geliştirme yöntemleri ve başarısına yönelik sorular yöneltilmiştir. Şekil 4.2'de izlendiği üzere % 69 katılımcı çevik yaklaşımlar için proje başarı oranlarının % 50'nin üzerinde olduğunu düşünmektedir.



Şekil 4.2. Çevik yöntemler ile gerçekleştirilen projelerin başarı oranı

Tablo 4.4'te yazılım geliştiricilerin çevik yaklaşımlarla ilgili görüşleri çeşitli ifadeler ile belirlenmeye çalışılmıştır. Anket katılımcılarının çoğu, çevik yaklaşımın verimliliği, kaliteyi ve müşteri memnuniyetini arttırdığını; ancak sadece bir arada çalışan küçük takımlar için uygun olduğunu belirtmektedir. Katılımcıların çoğu,

çevik yaklaşımlarda proje yönetiminin zor olduğu konusunda ortak görüş bildirmektedir.

Tablo 4.4. Katılımcıların çevik yöntemlere yönelik düşünceleri

	Katılıyor	Kararsız	Katılmıyor
	%	%	%
Çevik yaklaşımlar verimliliğimizi arttırdı.	43	23	34
Çevik yaklaşımlar üretilen sistemlerinin kalitesini arttırdı.	34	37	29
Çevik yaklaşımlar geliştirme maliyetini azalttı.	40	26	34
Çevik yaklaşımlar müşteri memnuniyetini arttırdı.	34	40	26
Çevik takımlar yeterli dokümantasyon hazırlamaz.	31	38	31
Çevik takımlar yeterli mimari tasarım yapmaz (address architecture) yoktur.	20	26	54
Çevik takımlar yeterli analiz yapmaz.	17	29	54
Çevik takımlar yeterli planlama yapmaz.	23	23	54
Çevik yazılım geliştirme disiplinsizdir.	14	20	66
Çevik yazılım geliştirme sadece geçici bir hevestir.	20	11	69
Çevik yazılım geliştirme sadece bir arada çalışan takımlara uygundur.	43	26	31
Çevik yazılım geliştirme sadece küçük takımlar içindir.	37	29	34
Çevik yazılım geliştirmenin sonucu düşük kalitedir.	11	29	60
Çevik yazılım geliştirme projelerini yönetmek zordur.	43	20	37

4.1.4. Geleneksel yazılım geliştirme anket soruları

Organizasyonlarında geleneksel yöntemleri kullandıklarını belirten katılımcılara geleneksel yöntemin avantaj ve dezavantajlarına yönelik sorular ile çevik yöntemleri kullanım isteklerine yönelik sorular sorulmuştur. Geleneksel yöntem kavramı ile belirtilmek istenen, çevik yöntemler dışında kalan organizasyonların kullandığı yazılım geliştirme süreci modelleridir. Çalışmanın bu bölümünde ise alınan yanıtlara ait bulgular paylaşılmaktadır.

Tablo 4.5'te görüldüğü üzere geleneksel yöntemleri kullanan katılımcıların büyük çoğunluğu (%42) kullandıkları yöntemin takip edilebilirlik özelliğini avantajlı özellik olarak vurgulamışlardır. Diğer taraftan yöntemin esnekliği ise az tercih edilen

(%16) özellikler arasında yer almıştır. Katılımcılara takip ettikleri yönteme ne tür bir özellik eklemek istedikleri sorulmuş; alınan yanıtlar çoğunlukla esneklik (%42) ve kullanılabilirlik (%25) özellikleri olmuştur. Bir yazılım geliştirme destek aracında olması gereken özellikler sorulduğunda esneklik katılımcıların büyük çoğunluğu (%40) tarafından en çok tercih edilen özellik olarak tanımlanmıştır.

Tablo 4.5. Geleneksel yöntem kullanılan proje bilgileri

	Esneklik	Kullanılabilirlik	Takip edilebilirlik	Öğretici bilgiler	Diğer
	%	%	%	%	%
Geleneksel yöntemin avantajlı özelliği	16	29	42	3	10
Geleneksel yönteme eklenmesi gereken özellik	42	25	8	8	17
Yazılım geliştirme aracındaki olması gereken özellik	40	24	28	4	4

Takip ettikleri yazılım sürecinin istemedikleri özellikleri olarak ise en fazla sürecin kolay kullanılabilir olmaması, süreç tekrarlarının desteklenmemesi, gereksiz dokümantasyon yapılması gibi konuları listelemişlerdir. Geleneksel süreci takip eden katılımcılara organizasyonlarının çevik yöntemleri ne kadar sürede adapte edebileceklerini düşündükleri sorulduğunda ise, adaptasyon süreci için en fazla ortalama 1 ila 2 yıl arasında bir süre tahmin etmişlerdir.

4.2. Görüşme Bulguları

Çalışma kapsamında gerçekleştirilen olay incelemesi görüşmelerine ait bulgular organizasyonlarda uygulanan çevik yazılım geliştirme kavramı ve uygulaması (KN_1), Scrum yazılım geliştirme yönteminin genel yapısı (KN_2), geleneksel yazılım geliştirme yönteminden Scrum yazılım geliştirme yöntemine geçiş (KN_3) ve organizasyonlarda kullanılan proje yönetim araçları (KN_4) temalarına ait kategoriler altında bu bölümde açıklanmaktadır. O1 ve O2 organizasyonlarında görev alan profesyonellerden alınan yanıtlar doğrultusunda elde edilen bilgiler ilgili başlıklar altında yer almaktadır.

4.2.1. Organizasyonlarda uygulanan çevik yazılım geliştirme kavramı ve uygulaması

Çevik yazılım geliştirme yöntemlerini organizasyonlarında kullanan profesyonellere organizasyonlarda uygulanan çevik yazılım geliştirme kavramı ve uygulamasına yönelik olarak sorular yöneltilmiştir. Bu sorulardan elde edilen bulgular daha çok çevik yöntem ile ilgili eğitimli personelleri olup olmadığı (KN_1_1), çevik yöntemin organizasyonlarında kullanım süreleri (KN_1_2), çevik yöntemin kullanıldığı proje sayısı (KN_1_3), çevik yöntem kullanan proje takımının büyüklüğü (KN_1_4) ve çevik yöntemin proje yönetimine sağladığı avantajlar, dezavantajlar (KN_1_5) konuları ile ilgilidir.

4.2.1.1. Çevik yöntem ile ilgili eğitimli personel

O1_SM ve O1_TU çevik yazılım geliştirme yöntemleri ile ilgili eğitimli personellerinin olduğunu belirtmektedirler. O2_SM ise Scrum ile ilgili yapılan PSM (Professional Scrum Master) sınavına girerek sertifika almaya çalıştıklarını ifade etmektedir.

4.2.1.2. Çevik yöntemin kullanım süresi

O1_SM bir senedir kurumda olduğunu ve organizasyona dahil olduğundan beri çevik yazılım geliştirme yöntemlerini kullandıklarını ve kurumda daha tecrübeli olan O1_TU ise çevik yazılım geliştirme yöntemlerinden Scrum yazılım geliştirme yöntemini 3 – 4 yıldır kullandıklarını belirtmektedirler. O2_SM ise organizasyonlarında 3 yılı aşkın bir süredir Scrum yazılım geliştirme yöntemini uyguladıklarını, daha önceleri ise geleneksel yazılım geliştirme yöntemlerinden Çağlayan yöntemini kullandıklarını bildirmektedir.

4.2.1.3. Çevik yöntemin kullanıldığı proje sayısı

O1_SM bulunduğu organizasyonda çevik yazılım geliştirme yöntemlerini kullandıkları dördüncü projede görev aldığını, O1_TU ise şimdiye kadar 7 – 8 çevik yazılım projesinde yer aldığını belirtmektedir. O2_SM organizasyonlarında tek bir projeleri olduğunu, aynı projede sürekli yeni sürüm çıkardıklarını ifade etmektedir.

4.2.1.4. Çevik yöntem kullanan proje takımının büyüklüğü

O1 içerisinde görev alan SM sadece bu takımda görev almaktadır. Takım içerisinde bir ürün sahibi, bir Scrum master ve ayrıca Scrum takım üyeleri bulunmaktadır. Takımın büyüklüğü yönetilen projeye göre farklılık göstermektedir. O1_SM organizasyonlarında genelde 7 - 8 kişilik takımlar oluşturduklarını, ancak bazı takımların 4 – 5 kişilik, bazılarının ise 12 – 13 kişilik olabildiğini belirtmektedir. O1_TU ise şimdiye kadar en fazla 13 kişilik takımlar halinde ve kapalı bir odada beraber çalıştıklarını ifade etmektedir. O2_SM ise iki Scrum takımlarının olduğunu, ortak olarak çalışan, biri 8 diğeri 7 kişilik olan bu takımlar içerisinde bir ürün sahibi, bir Scrum master, birer ekip lideri ve ayrıca Scrum takım üyelerinin bulunduğunu ve şimdiye kadar çalışan en büyük Scrum takımlarının ise 15 kişilik olduğunu ve Scrum takımlarının bir arada tek bir odada çalıştığını belirtmektedir.

4.2.1.5. Çevik yöntemin proje yönetimine sağladığı avantaj ve dezavantajlar

Çevik yazılım geliştirme yöntemlerinin organizasyonlara sağladığı avantaj ve dezavantajlar ile ilgili O1_SM organizasyonlarında Scrum'ın genel mantığı olan Scrum takımını bir arada ve tek odada çalıştırmanın kendilerini zorladığını, hiyerarşide proje bazlı organizasyon olmadığı ve projedeki her takım üyesi farklı bölüm müdürlükleri altında görev aldığı için bu özelliği uygulamanın kolay olmadığını ifade etmektedir. O1_TU ise çevik yazılımın kendilerine bir çok avantajlar sağladığını örneğin, Scrum'ın planlama sürecine kattığı esnekliği gözlemlediklerini belirtmektedir. O2_SM ise takip edilebilirlik özelliğinin Scrum yazılım geliştirme yönteminin öne çıkan bir özelliği olduğunu, analiz, dokümantasyon ve planlamalarının yeterli olduğunu ve Scrum yazılım geliştirme yöntemine geçtikten sonra iş yüklerinin azalarak, yaptıkları işin ölçülebilir hale geldiğini bildirmektedir.

4.2.2. Scrum yazılım geliştirme yönteminin genel yapısı

Görüşme yapılan profesyonellere organizasyonlarında kullandıkları Scrum yazılım geliştirme yöntemi içerisinde kullanılan Scrum rolleri (KN_2_1), Scrum pratikleri (KN_2_2), Scrum toplantıları süreleri (KN_2_3) ve Scrum kavramları (KN_2_4)

kullanımı ile ilgili kategoriler bu bölümde bildirilmektedir. Bölüm içerisinde profesyonellerin kendi sözleri ile anlatımlarına da yer verilmektedir.

4.2.2.1. Organizasyonlarda yer alan Scrum rolleri

O1_SM ve O1_TU ürün sahibi, takım üyeleri (analist, testçi, yazılım geliştirici) Scrum master dışında Scrum proje takımlarında bir de proje lideri kavramı olduğunu ve Scrum master'ın bu görevi de üstlendiğini belirttiler. O2 organizasyonu ise Scrum master, ekip lideri, ürün sahibi ve Scrum takım üyelerinden oluşmaktadır. O1_SM proje lideri ve Scrum master kavramlarının beraber kullanılmasını şu ifadelerle belirtmektedir; "Benim kimliğim Scrum master ve proje lideri. Ben proje lideri olarak birinci fazda çevik projeler içerisinde yer alıyorum, ikinci fazda Scrum master olarak görev alıyorum. Bir takımda hem Scrum master hem de proje lideriyim. Bizim kullandığımız Scrum mantığında bir proje lideri var. Çevik projelerde proje lideri isterse etkin rol alabilir. Bizde bu uygulanmaya çalışılıyor. Şube dönüşüm uygulamaları projesinde görev alıyorum şuan. Bu proje dışında da ben yine proje lideri kimliğimle çevik projelerde görev aldım".

4.2.2.2. Organizasyonlarda Scrum pratikleri kullanımı

O1_SM, O1_TU ve O2_SM ile yapılan görüşmeler sonucunda görev aldıkları Scrum takımlarının teoride Scrum modeli içerisinde yer alan günlük Scrum toplantısı, Sprint planlama toplantısı, Sprint değerlendirmesi (Review), geçmişe bakış (Retrospektif) etkinliklerini uygulama durumları Tablo 4.6'da gösterilmektedir. Organizasyonlarda yer alan Scrum pratikleri olarak adlandırılan tabloda, O1 ve O2 organizasyonlarına ait bilgiler yer almaktadır.

Tablo 4.6. Organizasyonlarda yer alan Scrum pratikleri

Kod	Günlük Scrum Toplantısı	Sprint Planlama Toplantısı	Sprint Değerlendirmesi (Review)	Geçmişe Bakış (Retrospektif)
O1	Yapılıyor	Yapılıyor	3 hafta sonunda yapılıyor.	Pratikte yok, ayrıca yapılmıyor, Sprint değerlendirmenin sonuna ekleniyor.
O2	Yok	Yapılıyor	Yapılıyor	Yapılıyor

İki Scrum takımı da teoride yer alan Sprint planlama toplantısını yaptıklarını, yine Sprint değerlendirmesi ve geçmişe bakış etkinliklerini takımlarında uyguladıklarını

ifade etmektedir. Scrum pratikleri arasında yer alan Sprint planlama toplantısında, profesyonellerin görüştükleri konular ve bu toplantılarda oluşturdukları Sprint iş listeleri ile ilgili detaylar görüşme içerisinde O1_SM tarafından şu ifadelerle anlatılmaktadır; "Sprint planlama toplantısında iş ihtiyaçlarını alıyoruz, neler yapılması gerekiyor, hangi talepler geldi, kontrol ediliyor, Ürün iş listesini oluşturacak maddeler oluşturuluyor. Planlama toplantısında görevleri belirliyoruz. 3 haftalık sprint içine kaç madde alınabilir diyerek Sprint iş listesini belirliyoruz. Sonra çalışmalar başlıyor. Kapasite planlamalarını da bu toplantıda yapıyoruz. Bu planlamayı yaptıktan sonra zaten kişilerin iş yoğunluğunu görebiliyoruz". O2_SM ise bu konu ile ilgili görüşlerini şu şekilde ifade etmektedir; "Sprint planlama toplantısında ürünlerin listelerini ve içeriklerini oluşturuyoruz. Mesela 10 ürün içeriğinden oluşan bir iş olduğunu düşünün. Toplantıyı Scrum master yönetiyor, planlama toplantısında ayrıca efor tahmini de yapıyoruz".

Geçmişe bakış Scrum pratiğine bakıldığında ise; O1 geçmişe bakışı ayrıca yapmayarak, sprint değerlendirmesinin sonuna eklerken, O2'de ise günlük Scrum toplantısı yerine, benzer şekilde haftalık Scrum toplantıları yapıldığı görülmektedir. Haftalık Scrum toplantısında da günlük Scrum toplantısına benzer şekilde "Bu hafta ne yapacağız, bir önceki haftadan eksiklerimiz var mı ve gelecek hafta için önümüzde engel var mı?" sorularının cevaplarını aradıklarını belirtmektedirler. O1 geçmişe bakışın ideal Scrum modelinde olup pratikte olmadığını, Sprint değerlendirme toplantısında yapılan hatalar dahil her şeyin konuşulduğunu ve bir sonraki sprint planlamaya dahil edilmeyecek olan adımları burada kararlaştırdıklarını söylediler.

O2 aynı Scrum takımı içerisinde 8 ve 7 kişilik iki grup oluşturduklarını, haftalık Scrum toplantılarına bu grupların ekip liderleri ile ayrıca ürün sahibi ve Scrum master'ın katıldığını ifade ettiler. 8 kişilik takım içerisinde sadece yazılım geliştiricilerin olduğunu, testçi ve analizcilerin ise 7 kişilik Scrum takımı içerisinde görev aldıklarını belirttiler. İdeal Scrum modelinde ise geliştirici takımı içerisinde yer alan tüm takım üyeleri analiz, yazılım ve test görevlerini gerçekleştirmektedir.

4.2.2.3. Organizasyonlarda Scrum toplantıları süreleri

Tablo 4.7'de Scrum takımlarının; günlük Scrum toplantısı Süresi, haftalık Scrum toplantısı Süresi, Sprint planlama toplantısı süresi, toplam Sprint süresi ve Sprint değerlendirme süreleri karşılaştırmalı olarak gösterilmektedir. O2_SM günlük Scrum toplantısı yerine, haftalık Scrum toplantıları yaptıklarını belirttiği için teoride yer almayan haftalık Scrum toplantısı etkinliği tabloya eklenmiştir. O1 günlük Scrum toplantılarında 15 dakikalık süreyi geçirmediğini belirtirken, O2 bu toplantılar yerine haftada bir, 1 saatlik haftalık Scrum Toplantısı yaptığını belirtmektedir. Her Sprint başında yapılan Sprint planlama toplantıları O1'de 2-3 saat aralığında bir süre alırken, O2'de bu toplantıların süresi 1-2 saat olarak değişmektedir. Toplam Sprint süresi O1'de 3 hafta iken O2'de bu süre 7 hafta olarak izlenmektedir. Sprint değerlendirme toplantısı süreleri O1'de maksimum 4 saat ile sınırlandırılırken, O2'de ise 1-2 saat aralığında değişmektedir.

Tablo 4.7. Organizasyonlarda yer alan Scrum pratikleri süreleri

Kod	Günlük Scrum Toplantısı Süresi	Haftalık Scrum Toplantısı Süresi	Sprint Planlama Toplantısı Süresi	Toplam Sprint Süresi	Sprint Değerlendirme Süresi
O1	15 dk	-	2-3 saat	3 hafta	Max 4 saat
O2	-	1 saat	1-2 saat	7 hafta	1-2 saat

4.2.2.4. Organizasyonlarda uygulanan Scrum kavramları kullanımı

Teoride Scrum modeli içerisinde yer alan Scrum kavramlarının, Scrum takımları içerisindeki kullanımlarına göre hazırlanan, beyaz tahta (Scrum Tahtası), ürün iş listesi (Product Backlog), kullanıcı hikayeleri (User Stories), iş bitim grafikleri (Burndown-Charts), engel içeriği (Impediment Backlog), bitti tanımı (Definition of Done) kavramlarının O1 ve O2 içerisindeki kullanım durumları Tablo 4.8'de belirtilmiştir. Organizasyonlarda yer alan Scrum kavramları olarak adlandırılan tabloda, O1 ve O2 organizasyonlarına ait bilgiler yer almaktadır.

Tablo 4.8. Organizasyonlarda yer alan Scrum kavramları

Kod	Beyaz Tahta	Ürün İş Listesi	Kullanıcı Hikayeleri	İş Bitim Grafikleri	Engel İçeriği	Bitti Tanımı
O1	Kullanılmıyor.	Yapılıyor	Yapılıyor	Kullanılıyor	Yapılmıyor	Yapılıyor
O2	Bazen kullanıyoruz	Yapılıyor	Yapılıyor	Kullanılıyor	Yapılıyor	Yapılıyor

O1 beyaz tahta kullanmadıklarını, proje yönetim aracındaki maddeleri perdeye yansıtarak izlediklerini belirtirken, O2 beyaz tahtayı zaman zaman kullandıklarını iletmektedir. İki organizasyonda da ürün iş listesi, kullanıcı hikayeleri, iş bitim grafikleri ve bitti tanımı kavramları kullanılırken, O2'de engel içeriği kavramının kullanılmadığı belirtilmektedir.

O2 içerisindeki bitti tanımı kavramı; ideal Scrum modelinden farklı olarak kullanılmaktadır. Sprint planlama toplantısı esnasında yapılan efor tahmininde, Scrum takım üyelerinin aynı iş için farklı puanlama yapmaları durumunda ortaya çıkmaktadır. Ürün sahibi Scrum takımında yer alan iki geliştiriciyi karşılaştırarak, farklı puanlamanın nedenini sorgulamaktadır. Sonuç olarak ortak bir karar ortaya çıkarılmaktadır. Bitti tanımı geliştirme takımı ve ürün sahibinin ortak karara ulaşması durumuna denilmektedir. O1'de ise bitti tanımı kavramı ideal Scrum modelinde olduğu gibi işin tam olarak bittiğinin takım üyeleri, ürün sahibi ve Scrum master tarafından belirlenmesi olarak kullanılmaktadır.

4.2.3. Geleneksel yazılım geliştirme yönteminden Scrum yazılım geliştirme yöntemine geçiş

Organizasyonlarında kullandıkları geleneksel yazılım geliştirme yöntemlerinden Scrum yöntemine geçişleri sırasında zorlandıkları durumlar (KN_3_2) ve geçiş süreleri (KN_3_1) ile ilgili kategorilere ait bilgiler bu bölümde yer almaktadır. O1 ve O2 organizasyonlarında görev alan profesyonellerden alınan yanıtlar doğrultusunda elde edilen bilgiler ilgili başlıklar altında yer almaktadır.

4.2.3.1. Geleneksel yöntemlerden Scrum yöntemine geçiş süresi

O1_SM, O1_TU ve O2_SM geleneksel yazılım geliştirme yöntemlerinden, Scrum yazılım geliştirme yöntemine geçişte organizasyonlarında harcanan süre bakımından ortak görüş bildirerek uyum sürecinin bir sene zaman aldığını belirtmektedirler. Çağlayan yönteminden yaklaşık 1 yıllık uyum süreci ile Scrum yazılım geliştirme yöntemini kullanmaya geçtiklerini ve bu yazılım geliştirme yönteminden memnun olduklarını iletmektedir. O1 içerisinde hala çağlayan yönteminin uygulandığı projelerde görev alan takım üyeleri bulunduğu ifade edilmiştir.

4.2.3.2. Geleneksel yöntemlerden Scrum yöntemine geçişte zorlanılan durumlar

O1_SM organizasyonlarında Scrum yazılım geliştirme yönteminden önce şelale yöntemini kullandıklarını, ancak geçiş esnasında büyük zorluklar yaşamadıklarını, Scrum pratiklerine ve kavramlarına kolaylıkla uyum sağladıklarını belirtmektedir. O1_TU ise Scrum yöntemine geçişlerinin birkaç yıl öncesine dayandığını, ayrıca organizasyonlarında hem şelale hem de Scrum yöntemlerini hala kullandıklarını ifade etmektedir. O2_SM ise Scrum pratikleri, kavramları ve araçları ile ilgili eğitim süreci dışında organizasyonlarında herhangi bir olumsuz durum yaşanmadığını, Scrum yazılım geliştirme yöntemine rahatlıkla uyum sağladıklarını, hatta Scrum'ın teoride yer alan özellikleri yerine pratikte kendileri için daha uygun olduğunu düşündükleri özellikleri eklediklerini belirtmektedir.

4.2.4. Organizasyonda kullanılan proje yönetim aracı

Görüşme yapılan profesyonellere organizasyonlarında Scrum yazılım geliştirme yöntemi için proje yönetim aracı kullanım durumları (KN_4_1), kullandıkları proje yönetim aracının genel özellikleri (KN_4_2), memnuniyetleri, eğer varsa araca eklenmesini istedikleri özellikler (KN_4_3) veya çıkartılmasını istedikleri özellikler (KN_4_4) ile ilgili sorular yöneltilmiştir. O1 ve O2 organizasyonlarında görev alan profesyonellerden alınan yanıtlar doğrultusunda elde edilen bilgiler ilgili başlıklar altında yer almaktadır.

4.2.4.1. Proje yönetim aracı kullanımı, genel özellikleri ve memnuniyet algısı

O1_SM ve O1_TU Scrum yazılım geliştirme yönteminde kullandıkları, TFS proje yönetim aracından genel olarak memnun olduklarını, bu aracın Scrum için uygun bir proje yönetim aracı olduğunu belirttiler. O2_SM ise Jira proje yönetim aracını kullandıklarını, aracın her türlü raporu kolayca alabilecekleri şekilde tasarlandığını ve gereksiz özelliklerinin bulunmadığını belirtmektedir.

4.2.4.2. Proje yönetim aracına eklenmesi ve araçtan çıkarılması gereken özellik

O1_SM ve O1_TU proje yönetim aracının eksik yönleri ile ilgili ise, iş akışlarının hızlandırıcı ve esnek yapıda olmadığını, ürün iş listesi içerisinde yer alan maddelerin statülerini güncellemek istediklerinde, uymak zorunda oldukları belli kuralların

işlerini zorlaştırdığını ve kendilerini yavaşlattığını belirttiler. Proje yönetim aracındaki bu karmaşık yapının raporlamalarında sorun olduğunu, sürecin daha basitleşmesi durumunda işin kolaylaşacağını belirttiler.

O2_SM organizasyonlarında Jira proje yönetim aracını kullandıklarını ifade eden profesyoneller, proje yönetim aracının eksik olan özelliği olarak ise, Jira'nın çağrı merkezi ile entegrasyonu olmamasının kendilerini olumsuz etkilediği durumlarla karşılaşabildiklerini belirtmektedir. O1_SM, O1_TU ve O2_SM'nin organizasyonlarında kullandıkları araçlar ve bu araçlara yönelik memnuniyetleri ile ilgili bilgiler Tablo 4.9'da yer almaktadır.

Tablo 4.9. Organizasyonlarda araç kullanımı

Kod	Araç kullanımı	Araç Memnuniyeti	Araca Eklenmesi Gereken Özellik	Araçtan Çıkarılması Gereken Özellik
O1_SM	TFS	Memnunum	Esneklik, test maddelerinin tek tek değerlendirilebilme özelliği	Süreç karmaşıklığı, prosedürel iş yükü, birden fazla test durumu girilmesi vb.
O1_TU	TFS	Memnunum	Raporlamalarda tamamlanan iş yüzdesi hatalı gösteriminin düzeltilmesi	Maddelerdeki statü değişikliği sorunu, alt maddelerde yer alan hata sebebi ile tüm maddenin hatalı olarak izlenmesi
O2_SM	Jira	Memnunum	Jira'nın çağrı merkezi entegrasyonu	Yok

4.3. Anket ve Görüşme Sonuçlarının Değerlendirilmesi

Yapılan anket çalışması kapsamında Türkiye'deki organizasyonların çevik yöntemleri ne oranda kullandıkları ve yazılım geliştiricilerin çevik yöntemlere yönelik düşünceleri belirlenmeye çalışılmıştır. Anket sonuçlarına göre, Türkiye'de çevik yöntemleri kullanan organizasyonların bu yöntemleri kullanma süreleri ortalama 2 yıldan fazladır; bu da Türkiye'de çevik yöntemlerin kullanımının çok yaygın olmadığını, ancak bu yöntemlerin kullanımına yönelik bir eğilim olduğunu göstermektedir. En yaygın olarak kullanılan yaklaşım olarak SCRUM ön plana çıkmaktadır. Sonuçlar Agile Türkiye'nin (Agile Türkiye, 2013) 2013 yılında gerçekleştirmiş olduğu çalışmanın sonuçları ve uluslar arası diğer çalışmaların sonuçları (URL-4, 2016) ile önemli oranda örtüşmektedir.

Çalışmaya katılan yazılım geliştiricilerin çevik yaklaşımlar konusundaki tutumları oldukça olumludur. Çevik yaklaşımların verimliliği, kaliteyi ve müşteri memnuniyetini arttırdığını düşünmektedirler. Ancak daha çok küçük takımlar için uygun olduklarını ve proje yönetiminin sağlanmasının zor olduğunu düşünenler de çoğunluktadır.

Görüşme sonuçlarına göre, O1 ve O2 organizasyonlarında uygulanan Scrum metodolojisi ile teoride yer alan Scrum yazılım geliştirme yönteminin farklı uygulanan özellikleri Tablo 4.10'da karşılaştırmalı olarak gösterilmektedir. Tablo detaylı olarak incelendiğinde, organizasyonların Scrum yazılım geliştirme yöntemini teoride belli olan özellikleri dışında, kendi ihtiyaçlarına uygun hale getirerek kullanmayı tercih ettikleri gözlemlenmektedir. Teoride tanımlanan ideal Scrum modelinde proje lideri kavramı yokken, O1'de bu Scrum master'ın görev tanımı içerisinde yer almaktadır. Günlük Scrum toplantısı ideal Scrum modelinde yer alırken, O2'de yer almamakta; ideal Scrum modelinde olmayan haftalık Scrum toplantısı ise O2'de yapılmaktadır. Geçmişe bakış O1'de Sprint değerlendirme toplantısı sonunda yapılırken, ideal Scrum tanımında yer alan beyaz tahta ve engel içeriği tanımları bu organizasyonda kullanılmamaktadır.

Tablo 4.10. O1 ve O2 organizasyonlarında Scrum özelikleri kullanımı

Scrum Özellikleri	İdeal Scrum Modeli	O1	O2
Proje Lideri	Yok	Var	Yok
Günlük Scrum Toplantısı	Var	Var	Yok
Haftalık Scrum Toplantısı	Yok	Yok	Var
Geçmişe Bakış	Var	Yok	Var
Beyaz Tahta	Var	Yok	Var
Engel İçeriği	Var	Yok	Var

Bu çalışma kapsamında yapılan görüşmelerden elde ettiğimiz analiz sonuçlarına göre; görüşülen profesyoneller geleneksel yazılım geliştirme yöntemlerinden Scrum yazılım geliştirme yöntemine geçişte büyük zorluklar yaşamamış, yeni yönteme ve içerisinde yer alan Scrum pratiklerine kolaylıkla uyum sağlamışlardır. Organizasyonlar, Scrum yazılım geliştirme yöntemini teoride belli olan özellikleri dışında, kendi organizasyonlarına uygun hale getirerek kullanmayı tercih etmektedirler. Organizasyonlardan birinde teoride yer alan geçmişe bakış toplantısı yapılırken, diğerinde sprint planlama toplantısı sonuna eklenmekte, ayrıca bir toplantı yapılmamaktadır. Yine O1 organizasyonunda günlük Scrum toplantısı

yapılırken, O2'de bu toplantılar yerine haftalık Scrum toplantısı yapıldığı belirtilmektedir. Scrum'ın teoride yer alan yapısına aykırı olan proje lideri kavramı O1 organizasyonunda görev tanımları içerisinde yer alırken O2 organizasyonu bunu kesinlikle reddetmektedir.

Görüşmelerin gerçekleştirildiği takım üyesi ve Scrum master'ların Scrum yazılım geliştirme yöntemi pratikleri ve araçları ile ilgili düşünceleri olumlu bir görünüm sergilemektedir. Görüşme katılımcıları, Scrum yazılım geliştirme yönteminin öne çıkan özelliklerinin takip edilebilirlik olduğunu, analiz, dokümantasyon ve planlamaların yeterli olduğunu belirtmektedir. Scrum'ın yapısı itibari ile planlamalarını kolaylaştırdığını bunun sonucunda Scrum yazılım geliştirme yöntemine geçtikten sonra yaptıkları işlerin ölçülebilir hale geldiğini, üzerlerindeki iş yükünü ve zamanı tahmin etmenin kolaylaştığını iddia etmektedir. Ayrıca organizasyonlarda Scrum yazılım geliştirme yönetimi için kullanılan proje yönetim araçlarından genel olarak memnun olduklarını, ancak araçların daha esnek bir yapıya sahip olmaları için gerekli düzenlemelerin yapılabileceğini ilettiler.

Görüşme sonuçlarına göre, personellerin sadece ilgili proje için ayrılmaması ve hiyerarşide proje bazlı organizasyon olmaması yüzünden, zaman zaman Scrum takım üyelerinin sürekli olarak aynı odada çalışamadıkları ve daha önce geleneksel yazılım geliştirme yöntemleri ile çalışan personellerin, Scrum yazılım geliştirme yöntemi içerisinde yer alan Scrum metodolojisi ve pratikleri ile çalışmakta zorlandıklarını, bu durumun ise Scrum takımını olumsuz etkileyebildiği belirtilmektedir. Ancak Scrum'ın temel özelliklerinden olan, Scrum takım üyelerinin aynı odada çalışması özelliğinin projelerin hızla ilerlemesi ve Sprint iş listesinde bulunan işlerin tamamlanması açısından faydalı olduğu belirtildiğinden Harmanlanmış Scrum modelinde de bu pratiğe oldukça dikkat edilmelidir.

Günlük Scrum toplantılarının günlük olarak yapılması ve ürün sahibi dahil olmak üzere projede yer alan her takım üyesinin katılması zorunluluğu kaynak planlaması açısından organizasyonları zorlamakta, Scrum takımı içerisinde yer alan personeli farklı bir proje içerisinde kullanma durumunu neredeyse ortadan kaldırmaktadır. Bu durumu video konferans ile aşmaya çalışan organizasyonlar ise, kimi zaman teknik

aksaklıklar, çoğu zamanda toplantı süresinin uzaması sebebi ile günlük toplantıları yapmamaktadır.

Scrum takım üyeleri içerisinde görev alan Scrum master, ürün sahibi ve takım üyelerine ek olarak proje lideri rolünün eklenmesi, takımın kendi kendini yönetirken açığa çıkabilecek sorunları elimine etmesine faydalı olabilir. Scrum modelini kullandığını belirten organizasyonlardan biri, proje lideri kavramını kabul etmezken, diğer organizasyonda proje lideri görev tanımının Scrum takımı içerisine kullanıldığı izlenmektedir.

Sonuç olarak, anket ve görüşme sonuçlarına dayanarak organizasyonlarında Scrum yazılım geliştirme yöntemini uyguladıklarını belirten profesyonellerden alınan görüşler doğrultusunda, gerçek dünyada organizasyonlarda uygulanan Scrum modeli ile İdeal Scrum modeli arasında bir çok farklı özelliklerin bulunduğu gözlemlenmiştir. Benzer ve aynı yönlerinin de olduğu gözlemlenen bu modeller, her organizasyon tarafından kendilerine özgü Scrum modeli olarak uygulanmaktadır. Organizasyonlar bu uygulamalardaki farklılıkları çoğu zaman tecrübeleri ile elde ettiklerini belirtmektedirler.

İdeal Scrum modeli ile organizasyonlarda uygulanan Scrum modellerinin, Scrum pratikleri ve kavramlarını uygulama durumları Tablo 4.11'de özetlenmektedir. İdeal Scrum modelinde var olan ve organizasyonlar içerisinde uygulanan Scrum modelinde de korunan kavram ve pratikler, sprint planlama toplantısı, sprint gözden geçirme toplantısı, ürün gereksinim listesi, sprint gereksinim listesi, sprint kalan zaman grafiği, Scrum takım üyelerinin aynı odada çalışma özelliği ve 4 haftalık Sprint süresi aşılmama kuralı olarak izlenmektedir.

İdeal Scrum modelinde yer alan ürün kalan zaman grafiği, geçmişe bakış ve günlük Scrum toplantıları ise organizasyonlarda uygulanan Scrum modelinde çoğu zaman yer almamaktadır. Toplantıların ayakta yapılması, İdeali Scrum modelinde yer alırken, bazı organizasyonlarda uygulanmakta, fakat bazı organizasyonlar da yer almamaktadır. Proje lideri rolü ve haftalık Scrum toplantısı kavramları ise İdeal Scrum modelinde yer almayan, fakat bazı organizasyonlarda uygulanan Scrum modellerinde izlenen özellikler arasında yer almaktadır.

Tablo 4.11. Organizasyonlarda Scrum pratikleri ve kavramları kullanımı

	İdeal Scrum Modeli	Organizasyonlar İçerisinde Scrum Modeli
Sprint Planlama Toplantısı	Var	Var
Sprint Gözden Geçirme Toplantısı	Var	Var
Geçmişe Bakış	Var	Sıklıkla yok
Günlük Scrum Toplantısı	Var	Sıklıkla yok
Haftalık Scrum Toplantısı	Yok	Bazı organizasyonlarda var
Ürün Gereksinim Listesi	Var	Var
Sprint Gereksinim Listesi	Var	Var
Sprint Kalan Zaman Grafiği	Var	Var
Ürün Kalan Zaman Grafiği	Var	Sıklıkla yok
Toplantıların ayakta yapılması	Var	Bazı organizasyonlarda var
Scrum takım üyelerinin aynı odada çalışması	Var	Var
Proje lideri rolü	Yok	Bazı organizasyonlarda var
4 haftalık Sprint süresi aşılmama kuralı	Var	Var

Görüşmelerden elde edilen sonuçlar, çalışma kapsamında çevik yazılım geliştirme yöntemleri ile ilgili gerçekleştirilen anket çalışmasının sonuçları ile büyük oranda örtüşmektedir. Ortak noktalar özetlenecek olursa, Scrum yazılım geliştirme yöntemine karşı algının olumlu olduğu ve organizasyonların bu yöntemi uygulamalarında kullandıkları belirtilmektedir. Geleneksel yazılım geliştirme yöntemlerinden Scrum yazılım geliştire yöntemine geçiş sonrasında, Scrum pratiklerinin yazılım geliştirme sürecindeki kaliteyi arttırdığı düşünülmektedir. İdeal Scrum modeli ile Türkiye'deki organizasyonlarda uygulanan Scrum modellerini karşılaştırma imkanı sağlayan Harmanlanmış Scrum modelinin, İdeal Scrum modelini organizasyonlarında uygulayacak profesyoneller için faydalı olacağı düşünülmektedir. Bu çalışma ile Scrum yazılım geliştirme yöntemini yeni kullanmaya başlayacak organizasyonlarda, bu yöntemi her yönüyle uygulamak zorunda oldukları bir model olarak görülmemesi, kendi sistemlerine uyarlayabilecekleri özellikleri içeren bir model olarak öngörülmesi hedeflenmektedir. Çalışma içerisinde farklı organizasyonlarda farklı şekillerde kullanılan Scrum yazılım geliştirme yöntemleri gösterilerek, Scrum yazılım geliştirme yönteminin çeşitli organizasyonlara uyarlanabilecek bir yapıda olduğu ifade edilmektedir.

5. HARMANLANMIŞ SCRUM MODELİ

Çalışma kapsamında çevik yöntemleri kullanan organizasyonlar incelendiğinde, anket ve görüşme veri toplama araçları ile elde edilen geri bildirimlere göre, her organizasyonun Scrum yazılım geliştirme yönteminin çeşitli özelliklerini, kavramlarını, pratiklerini ve sürecin uygulanma yöntemlerini kendi bünyesinde gelişen ihtiyaçları ve tecrübeleri doğrultusunda şekillendirmeye çalıştığı gözlemlenmektedir. İncelenen organizasyonların her biri kendi kullandıkları modeli Scrum yazılım geliştirme yöntemi olarak tanımlamakta, ancak modelin kimi özelliklerini aslına uygun olarak kullanırken, kimi özelliklerini uygulamalarına dahil etmemekte veya ideal süreç içerisinde yer almayan ancak kendi organizasyonlarına daha uygun olduğunu gözlemledikleri farklı özellikleri süreçlerine dahil etmektedirler.

1990'lı yılların başlarında Ken Schwaber tarafından tanımlanan modeli (Schwaber, 1995) İdeal Scrum Modeli olarak kabul edersek bu çalışma kapsamında daha çok organizasyonlarda pratikte uygulanan özelliklerini içerecek şekilde harmanlanmış Scrum çerçevesi önerilmektedir. İdeal Scrum Modeli ile Harmanlanmış Scrum modelinin birçok ortak noktası olduğu gibi, çalışma kapsamında gerçekleştirilen alan yazın araştırması, anket ve görüşmeler sonucunda elde edilen bulgular doğrultusunda önerilen modelde farklı süreçler ve özellikler de yer almaktadır.

İdeal Scrum modelinde bulunan sprint planlama toplantısı, sprint gözden geçirme toplantısı, ürün gereksinim listesi, sprint gereksinim listesi, Scrum takım üyelerinin aynı odada çalışma özelliği ve 4 haftalık Sprint süresi aşılamama gibi özellikler ise Harmanlanmış Scrum modelinin de temelini oluşturmaktadır. İdeal Scrum modelinde yer alan geçmişe bakış, günlük Scrum toplantısı ve toplantıların ayakta yapılması özellikleri ise harmanlanmış modelde kaldırılması önerilen pratikler arasındadır.

İdeal Scrum modelinde bulunmayan, fakat Harmanlanmış Scrum modeli içerisinde yer alan proje lideri rolü Scrum takım üyelerinin, kaynak kullanımlarının ve iş önceliklerini yöneticiliğini üstlenmekte, günlük Scrum toplantıları yerine ise İdeal

Scrum modelinde yer almayan haftalık Scrum toplantıları ve gerekli görülmesi durumunda yapılabilecek hafta arası Scrum toplantıları önerilmektedir. Harmanlanmış Scrum modelinde, İdeal Scrum modelinde yer alan geçmişe bakış toplantıları kaldırılarak, bu toplantılarda görüşülen konular sprint gözden geçirme toplantılarının içerisine dahil edilmekte, İdeal Scrum modelinde iki farklı şekilde gösterilebilen ürün ve sprint kalan zaman grafikleri ise, sadece ilgili Sprint'e ait işleri gösterecek şekilde tek bir grafik olarak tasarlanmaktadır. Harmanlanmış Scrum modelinde tanımlanan, her sprint sonuna opsiyonel olarak eklenebilecek, Sprint gözden geçirme zamanı ise yine İdeal Scrum modeli içerisinde yer almayan uygulamalardan biridir.

5.1. Harmanlanmış Model ve İdeal Scrum Modeli Farkları

Bu çalışmanın amacı, Scrum yazılım geliştirme yöntemini pratik olarak farklı organizasyonlarda uygulamış ve halen bu organizasyonlarda görev alarak uygulamalarına devam eden profesyonellerin bakış açıları ile İdeal Scrum modelinin organizasyonlar içerisinde yer alamayan veya ideal modelde bulunmadığı halde tecrübeler doğrultusunda profesyoneller tarafından kullanılmaya başlanan bazı özelliklerin tek bir model içerisinde birleştirilmesini sağlamaktır. Anket ve görüşme sonuçlarına dayanarak, oluşturulan bu model, farklı profesyonellerin ve organizasyonların pratikteki uygulamaları ve ideal Scrum modeli olarak belirtilen ve teoride tanımlanan süreçte yer alan özellikleri birlikte barındıran bir süreç tanımlaması nedeniyle Harmanlanmış Scrum Modeli olarak adlandırılmıştır.

Çalışma kapsamında önerilen Harmanlanmış Scrum Modeli ile literatürde teorik olarak tanımlanan İdeal Scrum Modeli arasındaki farklar Scrum rolleri, Scrum araçları, Scrum pratikleri, Scrum toplantıları ve Scrum süreci başlıkları altında açıklanmaktadır. Bu başlıklar altında İdeal Scrum modeli içerisinde yer alan tanımlamalar belirtilerek, harmanlanmış model içerisinde sürece eklenen veya çıkartılan özellikler ile sürecin farklı aşamaları gösterilmeye çalışılmaktadır.

İdeal Scrum modelinde yer alan pratiklerin, kavramların ve kuralların bazıları çalışma kapsamında elde edilen bulgulara dayalı olarak önerilen modelde korunurken bazılarının ise uygulanmaması öngörülmektedir. İdeal Scrum Modeli ve önerilen,

Harmanlanmış Scrum Modeli tanımlamaları arasındaki temel farklar Tablo 5.1'de görülebilmektedir.

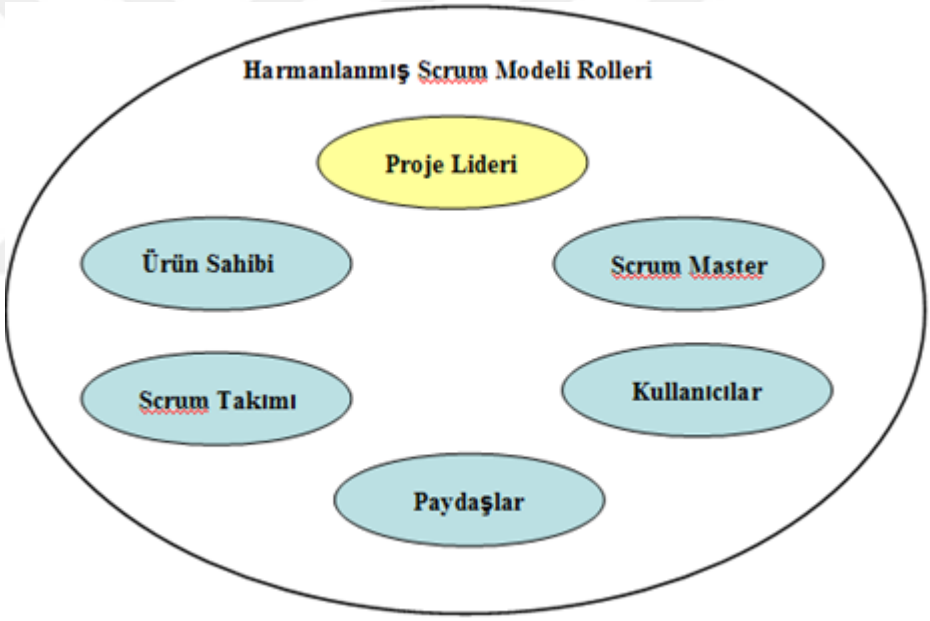
Tablo 5.1. Organizasyonlarda Scrum modeli karşılaştırmaları

	İdeal Scrum Modeli	Harmanlanmış Scrum Modeli
Sprint Planlama Toplantısı	Var	Var
Sprint Gözden Geçirme Toplantısı	Var	Var
Geçmişe Bakış	Var	Yok (Sprint gözden geçirme içerisine dahil)
Günlük Scrum Toplantısı	Var	Yok
Haftalık Scrum Toplantısı	Yok	Var
Ürün Gereksinim Listesi	Var	Var
Sprint Gereksinim Listesi	Var	Var
Sprint Kalan Zaman Grafiği	Var	Var
Ürün Kalan Zaman Grafiği	Var	Yok
Toplantıların ayakta yapılması	Var	Yok
Scrum takım üyelerinin aynı odada çalışması	Var	Var
Proje lideri rolü	Yok	Var
4 haftalık Sprint süresi aşılmama kuralı	Var	Var
Sprint Gözden Geçirme Zamanı	Yok	Var

Tablo 5.1'de izlendiği üzere, İdeal Scrum modelinde var olan ve Harmanlanmış Scrum modelinde de korunan kavram ve pratikler, sprint planlama toplantısı, sprint gözden geçirme toplantısı, ürün gereksinim listesi, sprint gereksinim listesi, Scrum takım üyelerinin aynı odada çalışma özelliği ve 4 haftalık Sprint süresi aşılmama kuralıdır. İdeal Scrum modelinde ürün kalan zaman grafiği ve sprint kalan zaman grafiği yapıları ise ayrı ayrı gösterilen grafikler yerine tek bir grafik ile ilgili Sprint'e ait kalan işleri gösterecek şekilde tasarlanmıştır. Günlük Scrum toplantısı ve bu toplantıların ayakta yapılması, İdeali Scrum modelinde yer alırken, Harmanlanmış Scrum modelinde ise uygulanmamaktadır. Aynı şekilde İdeal Scrum modelinde yer alan geçmişe bakış toplantıları, Harmanlanmış Scrum modelinde ayrıca yapılmayarak, sprint gözden geçirme toplantılarının içerisine dahil edilmiştir. Proje lideri rolü, sprint gözden geçirme zamanı ve haftalık Scrum toplantısı kavramları ise İdeal Scrum modelinde yer almayan, fakat Harmanlanmış Scrum modelinde yer verilen özellikler arasında yer almaktadır.

5.2. Harmanlanmış Scrum Modeli Rolleri

Harmanlanmış Scrum modelinde, proje lideri, ürün sahibi, Scrum takım üyeleri ve Scrum master olmak üzere temelde 4 rol bulunmaktadır. İdeal Scrum modelinde olduğu gibi gerekli durumlarda paydaşlar ve kullanıcılar yardımcı rolleri de Scrum rolleri içerisinde görev alabilmektedir. İdeal Scrum modelinde Scrum takım üyelerinin bir yöneticisi bulunmazken, harmanlanmış Scrum modeli içerisinde takım üyelerini yöneten bir proje lideri rolü yer almaktadır. Proje lideri rolü İdeal Scrum modelindeki Scrum master'ın bazı sorumlulukları ile geleneksel yazılım geliştirme modellerindeki proje lideri rollerinin aynı kişi üzerinde toplanmasını içermektedir. Harmanlanmış Scrum modeli içerisinde yer alan rollerin şekil üzerinde gösterimi Şekil 5.1'de izlenmektedir.



Şekil 5.1. Harmanlanmış Scrum modeli içerisinde yer alan roller

Tanımlanan proje lideri rolü Scrum takım üyelerini yönetirken, Scrum sürecinin yönetimi ve Scrum pratiklerinin kontrolü ise yine İdeal Scrum modelinde olduğu gibi Scrum master'a aittir. Proje lideri ekip içerisinde yer alan kaynakların kullanımı, aynı kaynağın birden fazla projede yer alması durumunda iş önceliklerinin belirlenmesi gibi işlerde görev alırken, Scrum master ise Scrum takımının Scrum pratiklerine ve kavramlarına uygun hareket ettiğini kontrol etmelidir. Sürecin doğru olarak işlemesi ve çevik yöntemlere uygun olarak devam etmesi adına her tür işlem Scrum master'ın sorumluluk alanındadır. Proje lideri ve Scrum master görev tanımı açısından

birbirlerine üstün olmak yerine, koordine bir şekilde giderek takım çalışmasına uyum sağlamalıdırlar. Proje liderinin Scrum takımı yönetimi çözüm odaklı olmalı, özellikle sorun yaşanan durumlarda takımın bir yöneticisi olduğu fark edilmelidir. Lider efor tahmini esnasında takım üyelerinin gerçek eforlarını ve yapabileceklerini belirttikleri işleri gözlemleyerek, gerekli durumlarda müdahale etmeli ve takım içerisinde hangi modülün kim tarafından yapıldığını bilmelidir. Olağan durumlarda sadece izleyerek görevini yapmalı, işlere gereksiz yere müdahil olmamalı, iş paylaştırmamalı ve Scrum sürecine uymayacak raporlamalar talep ederek takım üyeleri üzerinde baskı kurmamalıdır. Sorun yaşanması durumunda ortaya çıkmalı, duruma müdahil olarak takımda bir hakem gibi davranmalıdır. Proje lideri bir ekip değil, kişi olmalıdır. Gerekirse takım içerisinde yer alan yazılımcı, analistçi gibi takım üyelerinden biri olabilir, ancak ürün sahibi veya Scrum master olmamalıdır.

İdeal Scrum modelinde olduğu gibi, Scrum proje takımında, müşterinin isteklerini proje ekibine anlatan kişi ürün sahibidir, müşteri ile Scrum proje ekibi arasındaki etkileşimi sağlamakla görevlidir ve ortaya çıkarılacak ürünü yönetmekle yükümlüdür. Ürün sahibi, ürün iş listesinde yer alacak gereksinimleri ve kullanıcı hikayelerini belirlemede önemli rol oynamaktadır. Scrum takım üyeleri ise sürekli iletişim halinde olarak çalışan analiz, yazılım ve test safhalarının tümünde görev alabilen çalışanlardan oluşmaktadır. İdeal Scrum modelinde yer alan Scrum takımının kendi kendini yönetmesi özelliği, harmanlanmış Scrum modelinde Scrum rolleri arasına eklenen proje lideri rolü ile yerini gerektiğinde yönetilebilen Scrum takımı kavramına bırakmaktadır. Harmanlanmış modelin bu özelliği çevik projelerde uygulanan ideal Scrum rolleri arasında farklı bir yöntem olarak öne çıkmaktadır.

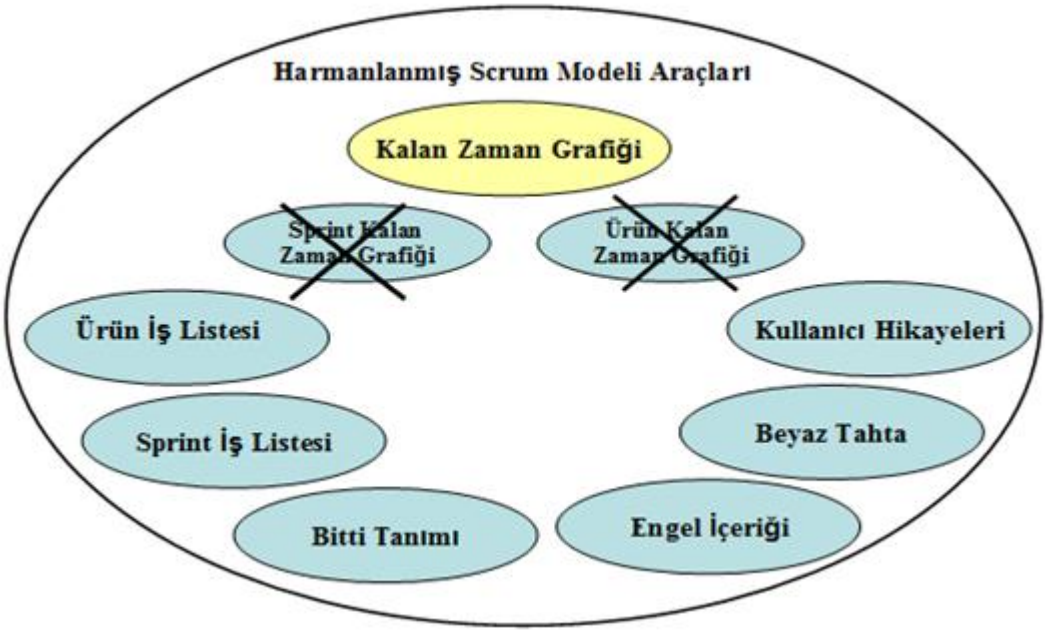
Organizasyon içerisinde yer alan faaliyetlerden etkilenen kişilerin veya sistemlerin oluşturduğu paydaşlar da gerekli durumlarda Scrum takımı içerisine dahil edilen roller arasında yer almaktadır. Paydaşlar olarak adlandırılan bu rol adı altında, şirket sahipleri, yöneticileri, çalışanları, tedarikçileri veya işçileri bulunabilirler. Bir organizasyon içerisinde yapılacak çalışmalarda, o organizasyonu tanıyan ve tiyaçlarını en iyi şekilde analiz edebilen kişilere ihtiyaç olabilmektedir. Nihai ürünü kullanacak olan kullanıcılarda Scrum takımı içerisinde mümkün olduğunca yer verilmesi gereken roller arasında yer almaktadır. Kullanıcılar rolü altındaki kişiler, geliştirme sırasında takıma çok dahil olmasalar da, gereksinim analizinde ve ürün

ortaya çıkarıldıktan sonra, özellikle son kullanıcı testlerinde önemli katkı sağlayabilmektedir. İdeal Scrum modelinde olduğu gibi Harmanlanmış Scrum modelinde de paydaşlar ve kullanıcılar Scrum takımının içerisinde gerekli durumlarda görev alabilmektedirler.

Çevik projelerde yer alan Scrum takım üyelerinin mümkün olduğunca aynı odada proje ekibi ile beraber çalışması özelliğinin etkileşimi arttırması (Abrahamsson, Ronkainen, Salo ve Warsta, 2002) sebebi ile bu özellik harmanlanmış Scrum modelinde de korunmaya çalışılmıştır. Scrum takımının genel olarak, çalışma disiplini olan, motivasyonu yüksek ve iletişim becerisi kuvvetli kişilerden oluşmasının sağlanması ise harmanlanmış modelde de öne çıkan özelliklerdendir.

5.3. Harmanlanmış Scrum Modeli Araçları

İdeal Scrum modeli içerisinde yer alan araçlar kullanıcı hikayeleri, ürün iş listesi, sprint iş listesi, engel içeriği, beyaz tahta, ürün kalan zaman grafiği, sprint kalan zaman grafiği ve bitti tanımı kavramları olarak listelenebilir. Bunların bir çoğu Harmanlanmış Scrum modelinde de işlevlerine uygun olarak kullanılmaktadır. Harmanlanmış Scrum modeli içerisinde yer alan araçların şekil üzerinde gösterimi Şekil 5.2'de izlenmektedir.



Şekil 5.2. Harmanlanmış Scrum modeli içerisinde yer alan araçlar

Scrum toplantıları beyaz tahta (Scrum tahtası) kullanılarak yapılarak, proje içerisindeki tüm işler ürün iş listesine yazılmakta, her sprint için sadece o yineleme içerisine dahil olacak işler sprint iş listesine alınmaktadır. Proje içerisinde yapılacak tüm işlerin kayıt edildiği ürün iş listesi içerisinden seçimler yapılarak oluşturulan, sprint iş listesinde yer alacak maddelerin mutlaka ürün iş listesi içerisinde belirlenmiş olması beklenmektedir. Proje sürecinde belirlenebilen engeller, Scrum master tarafından engel içeriği olarak tanımlanmakta, müşteriden alınan gereksinimler doğrultusunda çeşitli kullanıcı hikayeleri oluşturulmakta, bitti tanımı ile kullanıcı hikayelerinin tamamlanma kontrolü yapılmaktadır.

İdeal Scrum modelinde kimi zaman ürün kalan zaman grafiği ve sprint kalan zaman grafiği olarak ayrı ayrı tanımlanabilen grafiğin, karışıklığı engellemek adına harmanlanmış Scrum modelinde tek bir grafik ile gösterilmesi uygun görülmüştür. Harmanlanmış Scrum modelindeki sadece sprint kalan zaman grafiği ile gösterim yapılacak, sürekli gereksinim eklenebilen ürün kalan zaman grafiğine ise model içerisinde yer verilmeyecektir. Her sprint için sadece o Sprint iş listesi içerisinde yer alan işler yapılacağından dolayı, bu şekilde gösterimin sprint içerisindeki ilerleyişi görebilmek adına yeterli olacağı düşünülmektedir.

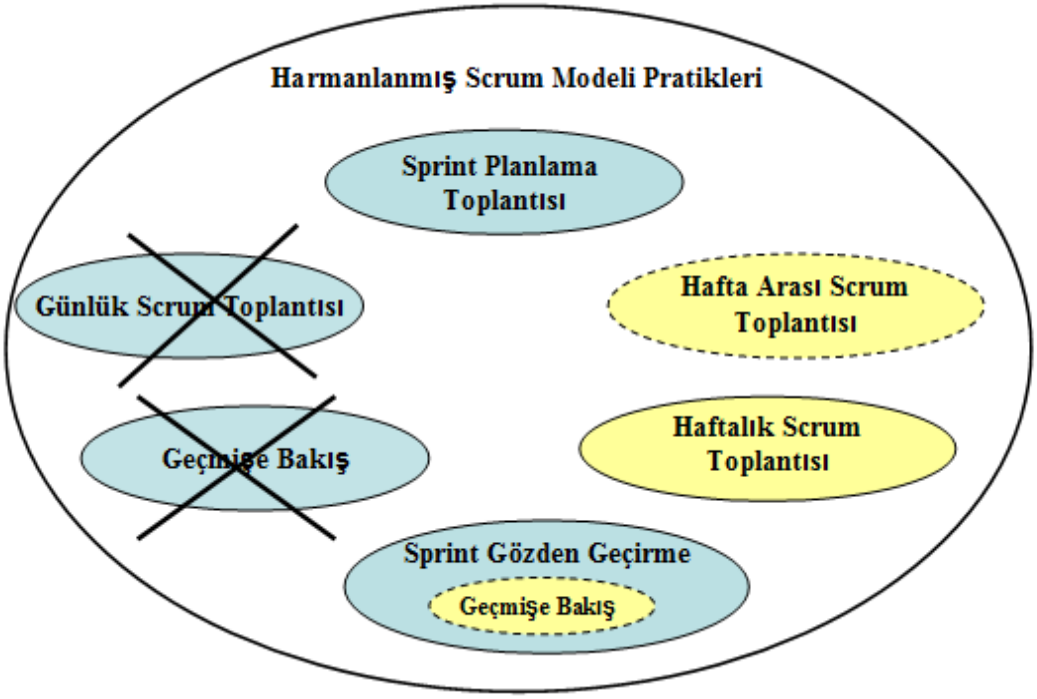
5.4. Harmanlanmış Scrum Modeli Pratikleri

Harmanlanmış Scrum modeli pratikleri iki bölümde incelenmektedir. Scrum toplantıları ve Scrum süreci olarak adlandırılan bölümlerde yer alan pratiklerin bazıları İdeal Scrum modeli içerisindeki görevlerine uygun olarak kullanılırken, bazıları çıkartılmış, bazı pratikler ise Harmanlanmış Scrum modeline ait özellikler ile güncellenmiştir.

5.4.1. Harmanlanmış Scrum modeli toplantıları

İdeal Scrum modeli içerisinde uygulanan pratikler sprint planlama toplantıları, günlük Scrum toplantıları, geçmişe bakış ve sprint gözden geçirme toplantılarıdır. Harmanlanmış Scrum modeli içerisinde uygulanacak Scrum pratikleri; Sprint planlama toplantıları, haftalık Scrum toplantıları, Sprint gözden geçirme toplantıları ve ihtiyaç duyulması durumunda bir kereye mahsus olarak yapılabilecek hafta arası Scrum toplantısı olarak tanımlanmaktadır. İdeal Scrum modeli ve Harmanlanmış

Scrum modeli içerisinde yer alan Scrum pratiklerinin farkları Şekil 5.3'te görülmektedir.



Şekil 5.3. Harmanlanmış Scrum modeli içerisinde yer alan pratikler

Sprint planlama toplantıları, İdeal Scrum modelinde olduğu gibi Harmanlanmış Scrum modelinde de her sprint başında yapılmaktadır. Tüm Scrum takım üyelerinin, Scrum master, ürün sahibi ve proje liderinin katıldığı bu toplantılarda proje içerisinde belirlenen ürün iş listesinde yer alan işler içerisinden mevcut Sprint'e dahil olacak işlerin seçimi ve bunlarla ilgili efor tahmini yapılmaktadır. Harmanlanmış Scrum modelinde Scrum takım üyeleri tarafından Sprint süresine dahil edilecek maddelerin alacakları eforu belirlemek için yapılan süre tahmini güncellemelerinde, proje liderinin görevi belirgin olarak ortaya çıkmaktadır. Proje lideri Scrum takım üyelerinin mevcut eforları ile yapabileceklerini belirttikleri işleri karşılaştırarak, gerekli durumlarda projenin sağlıklı ilerleyebilmesi adına müdahalelerini yapmalıdır. Sprint süresi yine İdeal Scrum modelinde olduğu gibi maksimum 4 hafta olarak sınırlandırılmalı, her Sprint kendi içerisinde analiz, tasarım, geliştirme, test ve teslim süreçlerini içermelidir.

İdeal Scrum modelinde yer alan günlük Scrum toplantılarının harmanlanmış Scrum modelinde haftalık Scrum toplantılarına yerine bırakmış olması ideal model içerisinde tanımlanan Scrum pratikleri içerisinde en çok değişkenlik gösteren yön

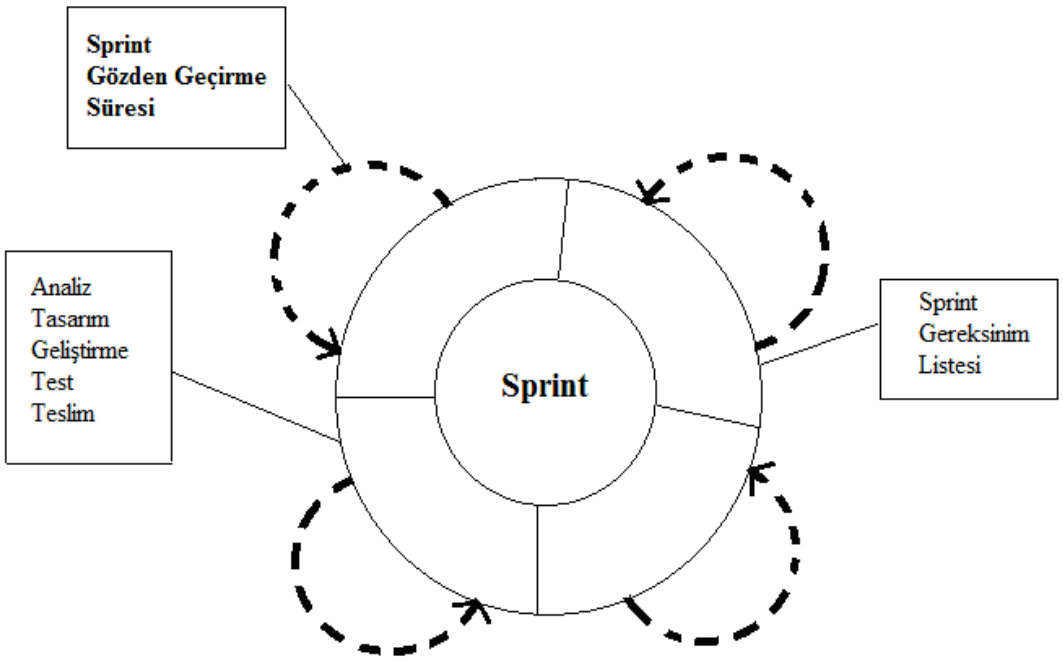
olarak dikkat çekmektedir. İdeal Scrum modelinde 15 dakikayı aşmayacak şekilde yapılan günlük Scrum toplantılarında, tüm proje ekibinde yer alan her takım üyesi "Dün ne yaptım?" "Bugün ne yapacağım?" "Karşılaşılan problemler ve yenilikler nelerdir?" sorularını cevaplamaktadır. Harmanlanmış Scrum modeli içerisinde günlük Scrum toplantılarına yer verilmemesinin nedenleri ise; 15 dakikalık kısa süresi, tüm takım üyelerinin cevaplaması gereken soruları yanıtlamasına fırsat vermemesi, takımın günlük zamanını planlanandan fazla şekilde işgal etmesi, bazı takım üyelerinin diğerlerine oranla daha uzun süre kullanması sebebi ile takım içerisinde anlaşmazlık oluşması ve beyaz tahta önünde ayakta olarak yapılmasının dikkat dağınıklığına sebep olması gibi etkenler ile özetlenebilmektedir. Bu toplantılar yerine bir saat süreli, toplantı odasında beyaz tahta kullanılarak ve oturarak yapılacak haftalık Scrum toplantılarının yapılması öngörülmektedir. Bu toplantıların süresinin günlük Scrum toplantılarına göre daha uzun olması sebebi ile ayakta yapılmasının verimi düşüreceği ve toplantıyı olumsuz etkileyeceği düşünülmektedir. Haftalık Scrum toplantılarının beyaz tahta önünde toplantı odasında yapılması ise, sürekli olarak aynı ortamda çalışan Scrum takımının etkileşimini arttırarak, toplantıya odaklanmayı sağlaması açısından faydalı olacağı düşünülmektedir. Bu toplantılarda günlük Scrum toplantılarına benzer şekilde "Geçen hafta ne yaptım?" "Bu hafta ne yapacağım?" "Karşılaşılan problemler ve yenilikler nelerdir?" sorularına cevap aranmalı, her Scrum takım üyesi bu sorular ile ilgili görüşlerini mümkün olduğunca eşit süreler alacak şekilde, takım arkadaşlarına aktarmalıdır.

Scrum takımı gerekli gördüğü durumlarda haftalık toplantıdan önce bir kereye mahsus olarak günlük Scrum toplantısını hafta ortasında yapabilir. Bu toplantı hafta arası Scrum toplantısı olarak adlandırılmaktadır. Hafta arası Scrum toplantısının yapılması zorunlu değildir, opsiyonel olarak Scrum takımı tarafından haftanın belirlenen bir gününde yapılabilir. Bu toplantılarda günlük Scrum toplantılarında cevaplanan sorulardan farklı olarak, takım içerisinde sorun oluşturan, beklenmedik şekilde ortaya çıkan veya Sprint içerisinde görüşülmesi gereken konular tartışılabilir. Bu özelliği ile günlük Scrum toplantılarından önemli ölçüde ayrılmaktadır. Hafta arası Scrum toplantıları, takım üyelerinin, Scrum master, ürün sahibi veya proje liderinin talebi ile gerçekleştirilebilir. Ancak bu toplantılar, bulunacağı günün günlük çalışma planını etkilememek açısından 15-20 dakikayı geçmemelidir.

4 haftalık Sprint süreci sonunda geliştirilen ürünün sunumunun yapıldığı, Sprint gözden geçirme toplantıları ise ideal Scrum modelinde yer alan, ancak görüşme sonuçları göz önüne alındığında organizasyonlarda çoğu zaman ayrı bir toplantı ile yapılmadığı tespit edilen geçmişe bakış toplantılarını da içinde barındıracak şekilde modellenmiştir. Scrum takım üyelerinin Sprint içerisinde doğru çalışan veya çalışmayan özellikleri tespit ettiği ve gelecek sprint için yapılacak değişiklikleri belirlediği geçmişe bakış ideal Scrum modelinde ekstra bir iş yükü olarak görülmektedir. Bu şekilde planlama ile geçmişe bakışında Scrum takımı içerisinde standart bir uygulama olarak görülmesi ve verimli hale getirilmesi hedeflenmektedir.

5.4.2. Harmanlanmış Scrum Sprint süreci

İdeal Scrum modeli mevcut Sprint yapılarında Sprint iş listesinde belirlenen maddelerin bir sonraki Sprint'e bırakılmaması ve maksimum Sprint süresi olan 4 haftanın geçirilmemesi, Scrum takım üyelerinden ekstra performans beklenmesine ve günlük çalışma sürelerinin artmasına sebep olmaktadır. Harmanlanmış Scrum modelinde, mevcut Sprint içerisinde hata çıkma olasılığı göz önünde bulundurularak hata ayıklamaya makul bir zaman ayrılmasının faydalı olabileceği düşünülerek, Sprint geliştirme safhasında 4 hafta sonunda yapılan Sprint gözden geçirme toplantısı sonrasında yer alacak, opsiyonel olarak uygulanabilen, Sprint gözden geçirme zamanı eklenmiştir. Maksimum 1 hafta sürecek şekilde uygulanacak bu zaman, Sprint iş listesinde tanımlanmış işlerin müşterinin isteği doğrultusunda olmadığı, başka bir deyişle ürünün kullanımının müşteri tarafından yeterince fonksiyonel ve performanslı bulunmadığı durumlarda kullanılabilecek ve Scrum takım üyelerine aynı Sprint içerisinde yer alan işleri gözden geçirme ve düzeltme imkanı sağlayabilecektir. Scrum takımının karar vereceği şekilde Sprint gözden geçirme süresi 1 günden başlayarak maksimum 1 hafta sürecek şekilde uygulanabilir. 4 haftalık sprint süresinin sonuna Sprint gözden geçirme süresinin eklenmesi ile maksimum kullanım süresi 4+1 olarak izlenmektedir. Opsiyonel olarak eklenen Sprint gözden geçirme süresinin gösterimi Şekil 5.4'te izlenmektedir. Ard arda yapılacak Sprint'lerin her birine bu süre opsiyonel olarak eklenebilir.



Şekil 5.4. Geliştirme safhasına Sprint gözden geçirme süresi eklenmesi

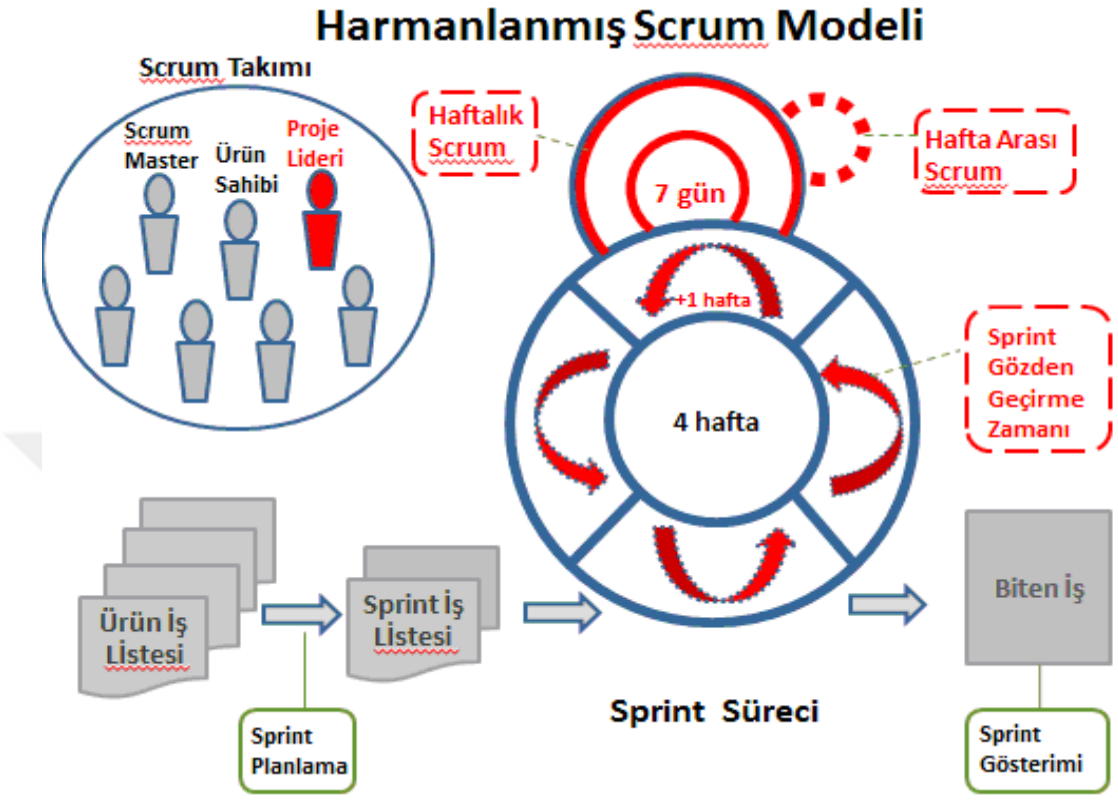
Sprint gözden geçirme zamanının kullanımı her Sprint için zorunlu olmamalı, Scrum takım üyeleri, Scrum master, proje lideri ve ürün sahibi verecekleri ortak karar doğrultusunda uygun görürlerse bu zamanı kullanmalıdırlar. Scrum takımı alacakları karara göre 1 gün veya maksimum 7 güne kadar bu zamanı kullanabilmelidir. Gerekli olması durumunda bu süre Sprint içerisinde yer alan ekstra durumlarda da kullanılabilmelidir.

İdeal Scrum modelinin genel yapısında bulunan ürün iş listesine sürekli yeni maddeler eklenebilmesi özelliğinin, projenin Sprint tekrarlarını uzatacağı ve proje süresinin uzatılmasının yönetim tarafından desteklenmediği durumlarda Sprint gözden geçirme zamanı kullanımının, ekstra Sprint tekrarı yapılmasına ve proje için yapılacak Sprint sayısını arttırmasına engel olabileceği düşünülmektedir. Sprint gözden geçirme süresi, 4 haftalık Sprint süresini aşmamak adına, Sprint içerisinde teslim edilecek iş sayısının azaltıldığı durumlarda kullanılarak, daha önce listeden çıkarılan maddelerin yapılmasını sağlayabilmektedir.

5.5. Harmanlanmış Scrum Modeli Gösterimi

Harmanlanmış Scrum modelinin genel yapısının şekil üzerinde gösteriminin yapıldığı bu bölümde, İdeal Scrum modelinde yer almayan ancak, Harmanlanmış Scrum modelinde yer alan pratikler ve kavramlara ait özellikler özetlenmektedir.

Çalışma kapsamında önerilen Harmanlanmış Scrum modeli ve Scrum pratiklerinin genel yapısı Şekil 5.5'te görülebilmektedir.



Şekil 5.5. Harmanlanmış Scrum modeli ve Scrum pratikleri

İdeal Scrum modelinde bulunmayan, fakat Harmanlanmış Scrum modeli içerisinde yer alan proje lideri rolü Scrum takım üyelerinin, kaynak kullanımlarının ve iş önceliklerini yöneticiliğini üstlenmektedir. Proje lideri Scrum takımı üzerinde baskı kurmamalı, üst yönetim ve Scrum takımı arasında gerekli koordinasyonu sağlamaya yönelik çalışmalarda bulunmalıdır. İdeal Scrum modelinde yer alan Scrum master kavramı ise, Harmanlanmış modelde de yer almakta, Scrum sürecinin yöneterek, Scrum pratiklerinin kontrolünü sağlamaktadır. Proje liderinin, ürün gereksinim listesindeki işleri belirleyen ürün sahibi ve Scrum sürecini yöneten Scrum master dışında bir kişi olması beklenmektedir.

Harmanlanmış Scrum modelinde sprint planlama toplantıları ve sprint gözden geçirme toplantıları dışında, günlük Scrum toplantıları yerine İdeal Scrum modelinde yer almayan haftalık Scrum toplantıları önerilmekte, ayrıca model içerisinde Scrum takım üyeleri tarafından gerekli görülmesi durumunda bir kereye mahsus olarak kullanılabilecek, hafta arası Scrum toplantıları da yer almaktadır. Opsiyonel olarak

yapılabilecek hafta arası Scrum toplantıları, İdeal Scrum modelinde bulunan günlük Scrum toplantılarında olduğu gibi, 15-20 dakikayı aşmayacak şekilde haftanın herhangi bir gününde yapılabilir. Harmanlanmış Scrum modelinde tanımlanan, her sprint sonuna opsiyonel olarak eklenebilecek, sprint gözden geçirme süresi ise 4 hafta süreli sprint sonunda belirlenen hataları ayıklamak için faydalı olabilecek, maksimum 1 hafta sürecek şekilde kullanılabilecektir.

Harmanlanmış Scrum modelinde sprint süreci içerisinde noktalı şekilde gösterilen alanlar, opsiyonel olarak kullanılabilecek alanlardır. İdeal Scrum modelinde de bulunan sprint planlama toplantısı, sprint gösteriminin yapıldığı sprint gözden geçirme toplantısı, ürün iş listesi, sprint iş listesi, 4 haftalık Sprint süresi özellikleri şekil üzerinde izlenmektedir. İdeal Scrum modelinde yer alan geçmişe bakış ve günlük Scrum toplantıları ise harmanlanmış model içerisinde yer verilmemektedir.

6. HARMANLANMIŞ SCRUM MODELİ UYGULANABİLİRLİĞİ

Harmanlanmış Scrum Modeli için Türkiye'deki yazılım organizasyonlarında görev alan ve doğrudan doğruya Scrum kullanımında deneyimli profesyonellere ulaşılarak, görüşlerinin alındığı değerlendirme sonuçları bu bölümde raporlanmaktadır. Çalışmanın daha önceki bölümlerinde anket ve görüşme sonuçları aracılığı ile çevik yazılım geliştirme yöntemlerinin organizasyonlar tarafından nasıl uygulandığı belirlenerek, İdeal Scrum modeli ve pratikte organizasyonlarda uygulanan Scrum modeli karşılaştırması yapılmıştır. Bu bulgular ile hazırlanan Harmanlanmış Scrum modelinin profesyonel kullanıma uygunluğunun belirlenmesi amacı ile bir değerlendirme hazırlanmıştır. Değerlendirme çalışması ile Harmanlanmış Scrum ve İdeal Scrum modelleri arasındaki genel fark ve benzerlikler maddeler halinde sunularak, Harmanlanmış Scrum modelinin uygulanabilirliği belirlenmek istenmiştir.

6.1. Harmanlanmış Scrum Takımı Rol ve Sorumlulukları

Harmanlanmış Scrum modeli Scrum takımı ve genel çalışma ilkeleri ile ilgili değerlendirme katılımcılarına yöneltilen konulara ait cevaplar bu bölümde raporlanmaktadır. Harmanlanmış Scrum modeline ait en önemli farklardan biri olan Scrum takım üyelerine proje lideri rolü eklenmesi özelliği, profesyonellerden alınan görüşlere göre detaylı olarak belirtilmektedir.

Değerlendirmeye katılan 26 katılımcının neredeyse tamamı Scrum takım üyelerinin sürekli iletişim halinde olmasını desteklerken, 17 kişilik bir kısım ise aynı odada çalışılması gerektiğini savunmaktadır. Katılımcıların 7 kişisi takım üyelerinin aynı odada çalışma fikrini benimsemezken, 2 kişi ise bu konuda kararsız kalmaktadır. 17 katılımcı Scrum takım üyelerinin, analiz, yazılım ve test safhalarının tümünde görev alabilen çalışanlardan oluşması kavramını desteklerken, 21 katılımcı ise Scrum takımının asli görevinin her sprint sonunda çalışan bir yazılımı ürün sahibine sunmak olduğunu düşünmektedir. Harmanlanmış Scrum takımında görev alan Scrum takım üyelerinin sorumlulukları ve çalışma şekillerine ait değerlendirme sonuçları detayları Tablo 6.1'de izlenmektedir.

Tablo 6.1. Harmanlanmış Scrum takımı çalışma şekli ve sorumlulukları

	Katılıyor	Kararsız	Katılmıyor
	#	#	#
Scrum takım üyelerinin sürekli iletişim halinde olması	25	0	1
Scrum takım üyelerinin aynı odada çalışması	17	2	7
Scrum takım üyelerinin, analiz, yazılım ve test safhalarının tümünde görev alabilen çalışanlardan oluşması	17	5	4
Scrum takımının asli görevinin her sprint sonunda çalışan bir yazılımı ürün sahibine sunmaktır	21	1	4

İdeal Scrum modelinden farklı olarak Harmanlanmış Scrum modeline eklenen proje lideri rolü ile ilgili, 12 katılımcının olumlu bir yaklaşım sergilediği gözlemlenmektedir. Katılımcıların 7 kişisi proje lideri rolünün eklenmesi fikrini kabul etmemekte yine kalan 7 kişi ise konu ile ilgili kararsız görüş bildirmektedir. Genel olarak proje liderinin eklenmesi fikrini benimsemeyen katılımcılar, üst yönetim ve Scrum takımı arasında koordinasyonu sağlamak, efor tahmini sırasında gerekli müdahaleleri yapmak, takım üyelerinin iş önceliklerini belirlemek, takım içerisinde hangi modülün hangi takım üyesi tarafından yapıldığını bilmek, takımın kaynak (örneğin personel) yönetimini sağlamak ve takım içerisinde yaşanan anlaşmazlıklara müdahale etmek konularında ise proje liderinin varlığını yüksek oranlarla benimsemektedir. Aynı değerlendirme katılımcıları proje lideri rolü ile Scrum takımının gerektiğinde yönetilebilen bir takım haline gelmesini ve bunun bir ekip ile değil bir kişi ile yapılmasını desteklemektedir. Hatta bu kişinin Scrum master, ürün sahibi, analist veya yazılım takım üyelerinden biri olabileceği konusunda görüşler bulunmaktadır.

Değerlendirme sonuçları, 21 katılımcının proje liderinin takım üyeleri üzerinde baskı kurmaması ve Scrum master ile koordine olarak çalışmasını gerektiğini gösterirken, 15 katılımcı ise takım içerisinde proje lideri tarafından iş paylaşımı yapılmasını istemediklerini belirtmektedir. Bu durum değerlendirme katılımcılarının Scrum takımı içerisindeki anlaşmazlıkları giderecek, koordinasyonu sağlayacak, kriz durumlarında sorumluluğu alarak, gerekirse takımı yönetebilecek bir lidere ihtiyaç duyduklarını, ancak bu kişinin takım içerisinde baskı kurmasını, Scrum takımının özgür çalışma disiplinini etkilemesini ve takım içerisinde yönetici kimliği ile iş

paylaşımı yapmasını desteklemediklerini göstermektedir. Proje lideri rolü ile ilgili değerlendirme sonuçlarına ait detaylar Tablo 6.2'de yer almaktadır.

Tablo 6.2. Proje lideri rolü sorumlukları

	Katılıyor	Kararsız	Katılmıyor
	#	#	#
Proje lideri üst yönetim ve Scrum takımı arasında gerekli koordinasyonu sağlamalı	22	1	3
Proje lideri efor tahmini sırasında gerekli durumlarda müdahale etmeli	13	6	7
Proje lideri Scrum takım üyelerinin iş önceliklerini belirlemeli	15	2	9
Proje lideri hangi modülün hangi takım üyesi tarafından yapıldığını bilmeli	19	2	5
Proje lideri ekip içerisinde kaynak (örn. personel) yönetimini sağlamalı	19	4	3
Proje lideri takım üyeleri arasında direk iş paylaşımı yapmamalı, takım içerisinde anlaşmazlık olduğu durumlarda müdahale etmeli	15	4	7
Proje lideri rolü ile Scrum takımı gerektiğinde yönetilebilen bir takım özelliği göstermeli	14	4	8
Proje lideri bir ekip değil, kişi olmalı	15	6	5
Proje lideri Scrum master ile koordine olarak çalışmalı	21	3	2
Proje lideri yazılımcı, analistçi gibi takım üyelerinden biri olabilir	13	3	10
Proje lideri ürün sahibi veya Scrum master olmamalı	8	9	9
Proje lideri takım üyeleri üzerinde baskı kurmamalı	21	2	3
Scrum takım üyelerine proje lideri rolü eklenmeli	12	3	11
Proje lideri yetki tanımı, Scrum master'ın bazı sorumlulukları ile geleneksel modeldeki proje liderlerinin görevlerini içermeli	12	7	7

Değerlendirme katılımcıları Harmanlanmış Scrum modelinde takım içerisinde yer alan Scrum master rolünün sorumlulukları ile ilgili Scrum sürecini yönetmesi ve takımın Scrum pratiklerine, kavramlarına uygunluğu kontrol etmesi konusunda olumlu görüş bildirmektedir. Katılımcılar ürün sahibi rolünün müşterinin isteklerini proje ekibine anlatan, müşteri ile Scrum proje ekibi arasındaki etkileşimi sağlayan ve ürün iş listesinde yer alan gereksinimler ile kullanıcı hikayelerini belirleyen kişi olduğu konusunda görüş birliği içindedirler. Katılımcıların 17'si organizasyonları

içerisinde yer alan faaliyetlerden etkilenen kişi veya sistemlerin oluşturduğu paydaşların gerektiğinde Scrum takımı içerisinde yer almaları onaylarken, sadece 10 tanesi son ürünü kullanacak kullanıcıların Scrum takımı içerisinde bulunmasını istemektedir. Scrum master, ürün sahibi, paydaşlar ve kullanıcılar rolleri ile ilgili değerlendirme sonuçlarına ait detaylar Tablo 6.3'te yer almaktadır.

Tablo 6.3. Scrum master, ürün sahibi, paydaşlar ve kullanıcılar rolleri

	Katılıyor	Kararsız	Katılmıyor
	#	#	#
Scrum master, Scrum sürecini yönetmeli ve Scrum pratiklerini kontrol etmeli	24	0	2
Ürün sahibi müşterinin isteklerini proje ekibine anlatmalı	21	2	3
Ürün sahibi, müşteri ile Scrum proje ekibi arasındaki etkileşimi sağlamalı	21	0	5
Ürün sahibi, ortaya çıkarılacak ürünü yönetmeli	21	2	3
Ürün sahibi, ürün iş listesinde yer alacak gereksinimleri ve kullanıcı hikayelerini belirlemeli	23	1	2
Organizasyon içerisinde yer alan faaliyetlerden etkilenen kişilerin veya sistemlerin oluşturduğu paydaşlar, gerekli durumlarda Scrum takımına dahil olmalı	17	3	6
Son ürünü kullanacak olan kullanıcılar gerekli durumlarda Scrum takımına dahil olmalı	10	7	9

6.2. Harmanlanmış Scrum Modeli Toplantıları

Harmanlanmış Scrum modeli içerisinde yer alan Scrum toplantıları ile ilgili değerlendirme katılımcılarının görüşlerine ait detaylar bu bölümde raporlanmaktadır. Harmanlanmış Scrum modelinde yer alan toplantılar ile ilgili görüşler bölüm içerisinde, Sprint planlama toplantısı, günlük, haftalık ve hafta arası Scrum toplantıları ve Sprint gözden geçirme toplantıları başlıkları altında incelenmektedir.

Harmanlanmış Scrum modeline ait en önemli farklardan biri olan İdeal Scrum modeli içerisinde yer alan sprint planlama toplantıları ve sprint gözden geçirme toplantıları Harmanlanmış Scrum modelinde de yer almaktadır. Günlük Scrum toplantıları yerine haftalık Scrum toplantıları yapılmakta; geçmişe bakış ise ayrıca yapılmayarak sprint gözden geçirme toplantıları içerisine dahil edilmektedir. İhtiyaç

duyulması durumunda bir kereye mahsus olarak yapılabilecek hafta arası Scrum toplantıları ise günlük Scrum toplantılarının yerini almakta, fark olarak haftada sadece bir kere yapılabilmektedir.

6.2.1. Sprint Planlama Toplantısı

Değerlendirme katılımcıları, Scrum takım üyeleri, Scrum master, ürün sahibi ve proje liderinin katılımı ile Sprint planlama toplantılarının her sprint öncesinde yapılması gerektiğini düşünmektedir. Katılımcıların tamamı Sprint planlama toplantılarında ürün iş listesinde yer alan işler içerisinden ilgili Sprint'e dahil olacak işlerin seçiminin ve efor tahmininin yapılmasını onaylamaktadır. Harmanlanmış Scrum modeli sprint planlama toplantısı ile ilgili değerlendirme sonuçlarına ait detaylar Tablo 6.4'te yer almaktadır.

Tablo 6.4. Harmanlanmış Scrum modeli sprint planlama toplantısı

	Katılıyor	Kararsız	Katılmıyor
	#	#	#
Sprint planlama toplantıları her sprint öncesinde yapılmalı	25	0	1
Sprint planlama toplantılarına, Scrum takım üyeleri, Scrum master, ürün sahibi ve proje lideri katılmalı	25	0	1
Sprint planlama toplantılarında, ürün iş listesinde yer alan işler içerisinden ilgili Sprint'e dahil olacak işlerin seçimi ve efor tahmini yapılmalı	26	0	0

6.2.2. Günlük, haftalık ve hafta arası Scrum toplantıları

Değerlendirme katılımcılarının tamamına yakını, günlük Scrum toplantılarının kaldırılması ile ilgili olumsuz görüş beyan ederken, bu toplantıların İdeal Scrum modelindeki hali ile ayakta yapılmasını onaylamaktadır. Katılımcıların yarısından fazlası günlük Scrum toplantıları yerine, "Geçen hafta ne yaptım?, Bu hafta ne yapacağım?, Karşılaşılan problemler ve yenilikler nelerdir?" sorularının cevaplanacağı bir saat sürecek haftalık Scrum toplantılarının yapılmasının uygun olmadığını düşünürken, kalan kısım kararsız veya olumlu olarak değerlendirmiştir. Harmanlanmış Scrum modelinde yer alan, haftanın belirli bir gününde, opsiyonel olarak yapılabilecek hafta arası Scrum toplantılarının yapılması ile ilgili katılımcıların yarısı olumlu görüş bildirirken, 13 katılımcı bu fikre olumsuz

bakmaktadır. Değerlendirme katılımcıları hafta arası Scrum toplantılarının 15-20 dakikalık süreyi geçmemesi konusunda görüş birliği içindedirler. Harmanlanmış Scrum modeli günlük, haftalık ve hafta arası Scrum toplantıları ile ilgili değerlendirme sonuçlarına ait detaylar Tablo 6.5'te yer almaktadır.

Tablo 6.5. Harmanlanmış model günlük, haftalık ve hafta arası Scrum toplantıları

	Katılıyor	Kararsız	Katılmıyor
	#	#	#
15 dakikalık günlük Scrum toplantıları yapılmamalı	4	2	20
Scrum toplantıları otururak yapılmalı	4	10	12
Günlük Scrum toplantıları kaldırılarak yerine bir saat sürecek haftalık Scrum toplantıları yapılmalı	6	2	18
Haftanın belirlenen bir gününde, opsiyonel olarak yapılabilecek hafta arası Scrum toplantısı yapılmalı	13	3	10
Hafta arası Scrum toplantısı Scrum takım üyelerinin, Scrum master, ürün sahibi veya proje liderinin talebi ile gerçekleştirilebilmeli	14	5	7
Hafta arası Scrum toplantıları 15-20 dakikayı geçmemeli	14	7	5

6.2.3. Sprint gözden geçirme toplantıları

Katılımcıların neredeyse tamamı sprint gözden geçirme toplantıları her sprint sonunda yapılması ve bu toplantılarda ürünün sunumunun yapılması konusunda hemfikir olarak izlenmektedir. Değerlendirmeye katılan katılımcıların önemli bir kısmı Harmanlanmış Scrum modelinde yer alan geçmişe bakış toplantılarının, Sprint gözden geçirme toplantıları içerisine dahil edilmesi özelliği ile ilgili olumlu görüş bildirmektedir. Harmanlanmış Scrum modeli Sprint gözden geçirme toplantıları ile ilgili değerlendirme sonuçlarına ait detaylar Tablo 6.6'da yer almaktadır.

Tablo 6.6. Harmanlanmış Scrum modeli sprint gözden geçirme toplantıları

	Katılıyor	Kararsız	Katılmıyor
	#	#	#
Sprint gözden geçirme toplantıları her sprint sonunda yapılmalı	24	0	2
Sprint gözden geçirme toplantısında ürünün sunumu yapılmalı	22	2	2
Geçmişe bakış toplantıları, ayrıca yapılmayarak Sprint gözden geçirme toplantılarının içerisine dahil edilmeli	10	9	7

6.3. Harmanlanmış Scrum Modeli Sprint Süreci

Katılımcıların önemli bir kısmı Harmanlanmış Scrum modelinde 4 haftalık Sprint süresinin aşılmaması ve Sprint iş listesinde belirlenen maddelerin bir sonraki Sprint'e bırakılmaması konusunda olumlu görüş bildirirken, her Sprint'in kendi içerisinde analiz, tasarım, geliştirme, test ve teslim süreçlerini içermesi gerektiğini desteklemektedir. Harmanlanmış Scrum modeline özgü özelliklerden biri olan Sprint içerisinde çıkabilecek hataları ayıklamaya makul bir zaman ayrılması ile ilgili katılımcıların 20'sinden olumlu geri bildirim alınmış, Sprint sonuna opsiyonel olarak kullanılabilen, Sprint gözden geçirme zamanı eklenmesi katılımcılar tarafından desteklenmiştir. Değerlendirme katılımcıları Sprint gözden geçirme zamanının Scrum takımı kararı doğrultusunda, 1 gün veya maksimum 7 güne kadar kullanılabileceği ve Scrum takım üyelerine Sprint içerisinde yer alan işleri gözden geçirme ve düzeltme imkanı sağlayabileceği fikirlerini benimsemektedir. Katılımcıların 10 tanesi Sprint gözden geçirme zamanı kullanımının ekstra Sprint eklenmesini engelleyebileceği görüşünü desteklemektedir. Harmanlanmış Scrum modeli Sprint süreci ile ilgili değerlendirme sonuçlarına ait detaylar Tablo 6.7'de yer almaktadır.

Tablo 6.7. Harmanlanmış Scrum modeli sprint süreci

	Katılıyor	Kararsız	Katılmıyor
	#	#	#
4 haftalık Sprint süresi aşılmamalı	16	4	6
Sprint iş listesinde belirlenen maddeler bir sonraki Sprint'e bırakılmamalı	15	5	6
Her Sprint kendi içerisinde analiz, tasarım, geliştirme, test ve teslim süreçlerini içermeli	10	9	7
Sprint içerisinde hata çıkma olasılığı göz önünde bulundurularak, hata ayıklamaya makul bir zaman ayrılmalı	20	3	3
Sprint süresi sonuna opsiyonel olarak kullanılabilen maksimum bir hafta süreli, Sprint gözden geçirme zamanı eklenmeli	11	8	7
Sprint gözden geçirme zamanı Scrum takımı kararı doğrultusunda, 1 gün veya maksimum 7 güne kadar kullanabilmeli	18	4	4
Sprint gözden geçirme zamanı, Scrum takım üyelerine aynı Sprint içerisinde yer alan işleri gözden geçirme ve düzeltme imkanı sağlamalı	18	4	4

Tablo 6.7. (Devam) Harmanlanmış Scrum modeli sprint süreci

	Katılıyor	Kararsız	Katılmıyor
	#	#	#
Sprint gözden geçirme zamanı, ekstra Sprint eklenmesini engelleyebilir	10	7	9
Sprint gözden geçirme zamanı, Sprint içerisinde teslim edilecek iş sayısının azaltıldığı durumlarda ek süre olarak kullanılmalı	11	5	10

6.4. Harmanlanmış Scrum modeli araçları

Değerlendirme katılımcılarının 20'den fazlası, Scrum projesi içerisindeki tüm işlerin ürün iş listesine yazılması, Sprint iş listesinde yer alacak maddelerin önceden ürün iş listesinde belirlenmiş olması ve her sprint için sadece o sprint'e dahil olacak işlerin sprint iş listesine yazılması ve ürün iş listesine proje süresi boyunca yeni maddeler eklenebilmesi konularında olumlu görüş bildirmektedir. Proje sürecinde belirlenebilen engellerin, Scrum master tarafından engel içeriği olarak kayıt edilmesi, müşteriden alınan gereksinimler ile kullanıcı hikayeleri oluşturulması, bitti tanımı ile kullanıcı hikayelerinin tamamlanma kontrolü yapılması ve Scrum toplantıları sırasında beyaz tahta kullanımı profesyoneller tarafından benimsenen Scrum araçları arasında yer almaktadır. Harmanlanmış Scrum modelinde, İdeal Scrum modelinden farklı olarak, ürün kalan zaman grafiği ve sprint kalan zaman grafiği yapıları ayrı ayrı kullanılmamakta, sadece Sprint kalan zaman grafiği kullanımının karışıklığı önlemek adına uygun olacağı düşünülmektedir. Bu yapı katılımcılara sunulduğunda 24 kişinin görüşlerinin olumlu olduğu izlenmiştir. Harmanlanmış Scrum modeli araçları değerlendirme sonuçlarına ait detaylar Tablo 6.8'de yer almaktadır.

Tablo 6.8. Harmanlanmış Scrum modeli araçları

	Katılıyor	Kararsız	Katılmıyor
	#	#	#
Beyaz tahta Scrum toplantıları sırasında kullanılmalı	22	3	1
Scrum projesi içerisindeki tüm işler ürün iş listesine yazılmalı	23	1	2
Her sprint için sadece o sprint'e dahil olacak işler, sprint iş listesine alınmalı	26	0	0
Sprint iş listesinde yer alan maddeler ürün iş listesinde belirlenmiş olmalı	26	0	0
Ürün iş listesine proje süresi boyunca yeni madde eklenebilmeli	21	2	3

Tablo 6.8. (Devam) Harmanlanmış Scrum modeli araçları

Proje sürecinde belirlenebilen engeller, Scrum master tarafından engel içeriği olarak kayıt edilmeli	25	1	0
Kullanıcı hikayeleri müşteriden alınan gereksinimler ile oluşturulmalı	25	1	0
Bitti tanımı ile kullanıcı hikayelerinin tamamlanma kontrolü yapılmalı	25	0	1
Ürün ve sprint kalan zaman grafikleri, ilgili sprint içerisindeki iş takibini kolaylaştırmak için sadece sprint'e ait işleri gösterecek şekilde düzenlenmeli	24	2	0

6.5. Harmanlanmış Scrum Modeli Değerlendirme Sonuçları

Harmanlanmış Scrum modelinin, Scrum yazılım geliştirme modelini kullanan organizasyonlarda uygulanabilirliğini belirleme amacı ile oluşturulan, değerlendirme maddelerinin yöneltildiği profesyonellerden elde edilen bulgulara göre; Scrum takım üyeleri İdeal Scrum modelinde olduğu gibi sürekli iletişim halinde olmalı ve aynı odada çalışmalıdır. Scrum takım üyeleri Harmanlanmış model içerisinde önerilen, gerektiğinde koordinasyonu sağlayarak, üst yönetimle verimli iletişim kurabilecek kriz durumlarında yönetici kimliğini ortaya çıkartabilecek bir proje liderine ihtiyaç duymaktadır; ancak proje lideri takım içerisinde baskı kurmamalı, iş paylaşımı yapmamalı kısacası Scrum takımının özgür çalışma disiplinini etkilememelidir. Sprint planlama ve sprint gözden geçirme toplantıları mevcut hali ile korunurken, geçmişe bakış toplantıları ayrıca yapılmayarak sprint gözden geçirme toplantıları içerisine alınmalıdır. Günlük Scrum toplantıları ise kesinlikle kaldırılmamalı, Harmanlanmış modelde önerilen haftalık Scrum toplantıları ise yapılmamalıdır.

Harmanlanmış modele özgü pratiklerden biri olan hafta arası Scrum toplantıları, takım içerisinde ihtiyaç duyulması durumunda Sprint içerisinde bir kereye mahsus opsiyonel olarak yapılabilmelidir. Her Sprint sonuna opsiyonel olarak kullanılabilen, 1 gün veya maksimum 7 güne kadar süreyi kapsayabilen, takım üyelerine Sprint içerisinde yer alan işleri gözden geçirme ve düzeltme imkanı sağlayan Sprint gözden geçirme zamanı eklenmelidir.

Harmanlanmış Scrum modeli içerisinde yer alan Scrum araçları katılımcılar tarafından genel olarak uygun bulunurken, ürün kalan zaman grafiği yerine sadece Sprint kalan zaman grafiği kullanılması da onaylanmıştır. Çalışmalar sonucunda

Harmanlanmış Scrum modelinin kimi özellikleri ve pratikleri profesyoneller tarafından benimsenmezken, bazılarının ise onaylandığı ve bu özelliklerin Scrum yazılım geliştirme yöntemini kullanan organizasyonlara uygun olduğu sonucuna ulaşılmıştır.

7. HARMANLANMIŞ SCRUM MODELİ İYİLEŞTİRİLMESİ

Çalışmanın bu bölümünde Türkiye'deki yazılım organizasyonlarında görev alan profesyonellerin görüşlerinin alındığı Ek-D içerisinde yer alan değerlendirme maddeleri bulguları doğrultusunda Harmanlanmış Scrum Modeli üzerinde yapılması gereken güncellemeler belirtilmektedir. Daha önceki bölümlerinde hazırlanan değerlendirme ile Harmanlanmış Scrum ve İdeal Scrum modelleri arasındaki genel farklar maddeler halinde belirlenerek, geri bildirimlerini almak üzere profesyonellere sunulmuştur. Elde edilen değerlendirme bulguları ile tekrar modellenen Harmanlanmış Scrum modeli profesyonel bakış açısı ile değerlendirilerek yeni bir görünüm kazanmıştır.

Harmanlanmış Scrum modelinin organizasyonlara uygunluğunu tespit etmek amacı ile hazırlanan değerlendirme sonuçlarının öne çıkan özelliği, İdeal Scrum modelinde var olan fakat Harmanlanmış Scrum modelinde kaldırılan günlük Scrum toplantılarının profesyoneller tarafından kaldırılmasının uygun görülmemesi sebebi ile tekrar modele eklenmesidir. Değerlendirmeye katılan 26 profesyonelin 18'i günlük Scrum toplantılarının kaldırılarak, yerine haftalık Scrum toplantılarının eklenmesi ile ilgili olumsuz görüş bildirmişlerdir. Ayrıca genel olarak Scrum toplantılarının oturarak yapılması ile ilgili alınan olumsuz geri bildirimler alınmıştır. Katılımcıların hafta arası Scrum toplantılarının eklenmesi ile ilgili olumlu görüş bildirmelerine rağmen günlük Scrum toplantılarının kaldırılmasını istememeleri bu konuya verdikleri önemi ve günlük Scrum toplantılarının Scrum'ın temel kavramları arasında yer aldığını düşündüklerini göstermektedir. Hafta arası Scrum toplantılarının eklenmesine olumlu bakılmasının sebebi olarak, günlük Scrum toplantılarında görüşülecek konuların belirli sorular üzerine kurulu olması, toplantı esnasında bu soruların cevaplarına odaklanarak başka konuların görüşülmemesi ve bu nedenlerle takım içerisinde yaşanan olumsuz durumlar veya görüşülmesi gereken farklı konular ile ilgili toplantı yapılamaması gibi nedenlerin olduğu düşünülmektedir. Değerlendirme katılımcıları Scrum takım üyelerinin, Scrum master, ürün sahibi veya proje liderinin talebi ile gerçekleştirilebilecek hafta arası

Scrum toplantılarını, takım içerisinde görüşülmesini istedikleri farklı konuları veya anlaşmazlıkları tartışabilecekleri serbest bir toplantı zamanı olarak görmüş ve modele eklenmesi ile ilgili olumlu görüş bildirmişlerdir. Açıklanan bu sebeplere dayanarak Harmanlanmış Scrum modelinin iyileştirmesinde zorunlu haftalık Scrum toplantıları kaldırılarak yerine günlük Scrum toplantıları dahil edilmiş, Scrum toplantıları ise ayakta yapılacak şekilde güncellenmiştir. Alınan olumlu geri bildirimlere dayanarak, opsiyonel olarak yapılabilecek hafta arası Scrum toplantılarına İyileştirilmiş modelde de yer verilmiştir.

Değerlendirme katılımcılarının 12 tanesi, Scrum takım üyelerine proje liderinin eklenmesi ile ilgili olumlu görüş bildirmişlerdir. Proje lideri rolünün eklenmesi fikrini kabul etmeyen veya konu ile ilgili kararsız görüş bildiren katılımcıların takım içerisinde yaşanan anlaşmazlıklara müdahale etmek, üst yönetim ve Scrum takımı arasında koordinasyonu sağlamak, efor tahmini sırasında gerekli müdahaleleri yapmak, iş önceliklerini belirlemek veya kaynak (örneğin personel) yönetimini sağlamak gibi konularda proje lideri kavramına sıcak baktıkları tespit edilmiştir. Değerlendirme bulgularından elde edilen sonuç, katılımcıların Scrum takımı içerisinde yukarıda belirtilen işleri yapabilecek bir yöneticiye ihtiyaç duydukları, ancak bu kişinin görev tanımının takım üzerinde baskı kurma veya yönetici kimliği ile takım üyeleri içerisinde direk iş paylaşımı yapma gibi özellikleri içermemesini istediklerini göstermektedir. Bu durum Harmanlanmış Scrum modeli içerisinde belirtilen proje liderinin görev tanımlarına uymasına rağmen, İyileştirilmiş modelde proje lideri rolünün yetkilerinin kesin çizgiler ile netleştirilmesi, takımı rahatsız edecek şekilde yönetici kimliği kullanımına izin verilmemesi olarak ortaya çıkarılmıştır. Proje liderinin eklenmesini isteyen katılımcıların olumlu görüşleri model içerisinde değerlendirilmiş, proje lideri rolü kaldırılmamıştır. Ancak bu konu ile ilgili kararsız veya olumsuz görüş bildiren katılımcıların görüşleri doğrultusunda proje liderinin yetkileri kısıtlanmış veya başka bir deyişle görev alanı net bir şekilde belirtilmiştir. Harmanlanmış modelin iyileştirilmesinde, proje liderinin öncelikle Scrum takımı içerisinde bir takım üyesi olduğu, kriz durumları haricinde, olağan durumlarda proje liderinin de, diğer takım üyelerinden farklı olmadığı belirtilmektedir. Takım üyeleri içerisinde yer alan proje lideri rolünün görev tanımı, kriz durumlarında Scrum master ile koordine olarak çalışmak, olağanüstü

durumlarda takımı yönetmek, Scrum takımı içerisinde yaşanabilecek anlaşmazlıkları gidermek, üst yönetim ve benzeri dış etkilere karşı sorumluluk almak, efor tahmini, iş önceliklendirme veya kaynak yönetimi konularında takım üyelerine destek olmak şeklinde güncellenmiştir.

Harmanlanmış Scrum modelinde yer alan Sprint süreci, Sprint planlama toplantısı, Sprint gözden geçirme toplantısı, Sprint gözden geçirme zamanı, Scrum araçları ve Scrum takım üyeleri ile ilgili olumlu geri bildirimler alınması sebebi ile, İyileştirilmiş modelde bu kavramlar ile ilgili herhangi bir değişiklik yapılmamıştır. Tablo 7.1'de Harmanlanmış Scrum modeli ve İyileştirilmiş model arasındaki temel farklar karşılaştırmalı olarak gösterilmektedir.

Tablo 7.1. Harmanlanmış Scrum modeli ve iyileştirilmiş model karşılaştırmaları

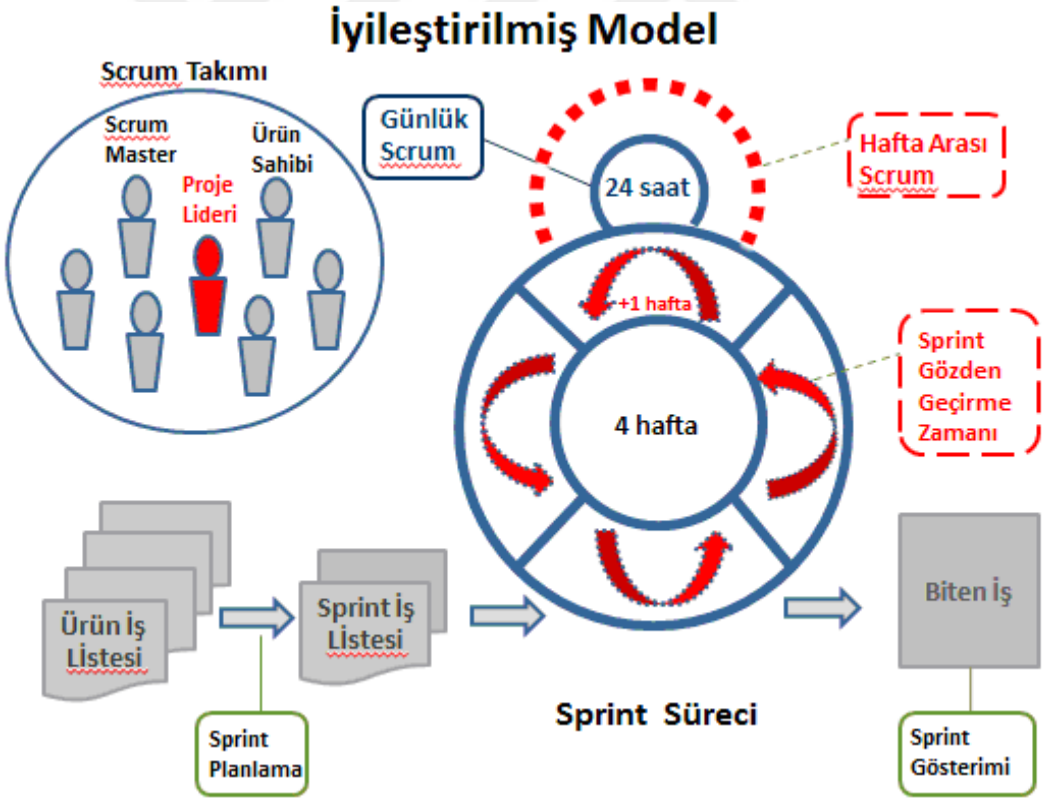
	Harmanlanmış Scrum Modeli	İyileştirilmiş Model
Sprint Planlama Toplantısı	Var	Var
Sprint Gözden Geçirme Toplantısı	Var	Var
Geçmişe Bakış	Yok (Sprint gözden geçirme içerisine dahil)	Yok (Sprint gözden geçirme içerisine dahil)
Günlük Scrum Toplantısı	Yok	Var
Haftalık Scrum Toplantısı	Var	Yok
Hafta Arası Scrum Toplantısı	Var	Var
Ürün Gereksinim Listesi	Var	Var
Sprint Gereksinim Listesi	Var	Var
Sprint Kalan Zaman Grafiği	Var	Var
Ürün Kalan Zaman Grafiği	Yok	Yok
Toplantıların ayakta yapılması	Yok	Var
Scrum takım üyelerinin aynı odada çalışması	Var	Var
Proje lideri rolü	Var	Var (Yetki daraltılması mevcut)
4 haftalık Sprint süresi aşılmama kuralı	Var	Var
Sprint Gözden Geçirme Zamanı	Var	Var

Tablo 7.1'de izlendiği üzere, Harmanlanmış Scrum modelinde yer alan ve İyileştirilmiş modelde de korunan kavram ve pratikler, sprint planlama toplantısı, sprint gözden geçirme toplantısı, sprint gözden geçirme zamanı, ürün gereksinim listesi, sprint gereksinim listesi, Scrum takım üyelerinin aynı odada çalışma özelliği, sprint kalan zaman grafiği ve 4 haftalık sprint süresi aşılmama kuralıdır. Her iki modelde de ürün kalan zaman grafiği kullanılmamak ve geçmişe bakış toplantıları, ayrıca yapılmayarak, sprint gözden geçirme toplantılarının içerisine dahil

edilmektedir. Haftalık Scrum toplantısı ve bu toplantılar oturarak yapılması, Harmanlanmış Scrum modelinde yer alırken, İyileştirilmiş modelde ise uygulanmamaktadır. Değerlendirme sonuçları kapsamında hazırlanan İyileştirilmiş modelde bu toplantıların yerini günlük Scrum toplantıları almakta ve toplantılar aslına uygun şekilde ayakta yapılmaktadır. Harmanlanmış Scrum modelinde yer alan proje lideri rolü ise İyileştirilmiş modelde de yer almakta, fakat görev tanımı aynı kalsa da yetkileri kısıtlanmaktadır.

7.1. İyileştirilmiş Model Gösterimi

İyileştirilmiş modelin genel yapısının şekil üzerinde gösteriminin yapıldığı bu bölümde, Harmanlanmış Scrum modeli üzerinde değişiklikler yapılarak profesyonellerden alınan geri bildirimlere uygun bir hale getirilmiştir. İyileştirilmiş modelde yer alan pratikler ve kavramlara ait özellikler Şekil 7.1'de özetlenmektedir.



Şekil 7.1. İyileştirilmiş model ve pratikleri

Şekil 7.1'de izlendiği üzere Harmanlanmış Scrum modelinde yer alan haftalık Scrum toplantıları İyileştirilmiş modelde yerini günlük Scrum toplantılarına bırakmıştır. Opsiyonel olarak yapılabilecek hafta arası Scrum toplantıları ve Sprint gözden

geçirme zamanı İyileştirilmiş modelde de yer alan pratikler arasındadır. Harmanlanmış Scrum modelinde yer alan proje lideri rolü İyileştirilmiş modelde de izlenmekte, ancak Scrum takım üyeleri arasındaki yeri farklı olarak konumlandırılmaktadır. Şekil üzerindeki bu değişiklik yetkilerine getirilen kısıtlamayı sembolize etmektedir.

8. SONUÇ ve ÖNERİLER

Bu çalışmada çevik yazılım geliştirme yöntemlerinden en yaygın olarak kullanımı belirlenen Scrum metodolojisinin organizasyonlarda kullanımı sırasında gözlemlenen Scrum ama pratiklerini engellemek için Harmanlanmış Scrum modeli önerilmektedir. Bu model ile Scrum'ın ana felsefesine aykırı olacak modelde yapılacak kontrolsüz değişikliklerin organizasyonlarda rastgele yapılması yerine, organizasyonlarda kullanımı kabul gören iyi pratikler ile modelin genişletilmesi sağlanmaktadır.

8.1. Yapılan Çalışmanın Özeti

Bu tez çalışması kapsamında Scrum yazılım geliştirme sistemini uygulayan organizasyonlarda kullanıma yönelik Harmanlanmış Scrum Modeli hazırlanmıştır. İdeal Scrum Modeli ile bir çok ortak noktası olan Harmanlanmış Scrum modeli, farklı pratikleri de içinde barındıran bir model olarak önerilmektedir. Çalışmanın başında çevik yazılım geliştirme ile ilgili bir anket hazırlanarak, organizasyonlarında Scrum kullanan profesyonellerin, çevik metodolojiler ve Scrum yazılım geliştirme modeli ile ilgili görüşleri alınmıştır. Anket çalışması daha sonra yapılan görüşmeler ile desteklenmiştir. Anket ve görüşme sonuçları, organizasyonların Scrum modelini alan yazında tanımlanan İdeal Scrum modeli pratikleri ve kavramları dışında, kendi organizasyonlarına uyarlayarak kullandıklarını göstermektedir. Alan yazında Scrum ama (Scrum but) olarak belirtilen bu durumu (Schwaber, 2011; Sutherland, 2009) destekleyen anket ve görüşmeler Harmanlanmış Scrum modelinin ortaya çıkışını sağlamıştır. Geliştirilen bu model bir değerlendirme ile görüşleri alınmak üzere organizasyonlarında Scrum kullanan profesyonellere sunulmuştur. Alınan geri bildirimler doğrultusunda Harmanlanmış Scrum Modeli üzerinde iyileştirmeler yapılarak, İyileştirilmiş Model oluşturulmuştur. Profesyoneller, Harmanlanmış model içerisinde yer alan proje lideri rolü, hafta arası Scrum toplantısı ve Sprint gözden geçirme süresi pratikleri ile ilgili olumlu görüş bildirerek, modelin Scrum kullanan organizasyonlarda uygulanabilirliği konusunda görüş bildirmişlerdir.

8.2. Harmanlanmış Scrum Modeli Özeti

Bu çalışma kapsamında önerilen Harmanlanmış Scrum modeli, Scrum yazılım geliştirme yöntemini organizasyonlarında uygulayan ve bu konuda deneyim sahibi olan profesyoneller ile gerçekleştirilen anket ve görüşmeler sonrasında geliştirilmiştir. Harmanlanmış Scrum modelinin ortaya çıkış amacı, Scrum yazılım geliştirme yönteminin, organizasyonlarda uygulanma şeklini tespit ederek, İdeal Scrum Modeli ile organizasyonların uyguladıkları Scrum modellerinin farklarını gösterebilmek ve bu şekilde Scrum metodolojisini teoride kalan özellikleri yerine, pratikte organizasyonlarda kullanılan özellikleri ile modelleyebilmektir. Anket ve görüşme bulgularına göre elde edilen, Scrum rolleri içerisinde proje lideri rolünün eklenmesi, hafta arası Scrum toplantıları ve Sprint gözden geçirme zamanının eklenmesi olumlu geribildirimler arasında yer alarak, modelin uygulanabilirliğini desteklemektedir. Harmanlanmış model içerisinde önerilen, günlük Scrum toplantılarının kaldırılarak, haftalık Scrum toplantılarının yerleştirilmesi ise profesyoneller tarafından ret edilerek, İyileştirilmiş modelin ortaya çıkmasını sağlamıştır. İyileştirilmiş model ile çalışmanın daha güvenilir ve uygulanabilir bir yapıya gelmesi hedeflenmiştir. Profesyonellerden alınan geribildirimler doğrultusunda, model üzerinde proje lideri rolünün yetkileri kısıtlanarak, Scrum takımının özgür çalışma disiplinini engelleyecek bir yönetici kimliğine izin verilmemesi sağlanmıştır.

8.3. Harmanlanmış Scrum Modelinin Katkıları

Bu çalışma ile önerilen Harmanlanmış Scrum modeli, Scrum yazılım geliştirme yönteminin organizasyonlarda uygulanan özellikleri ve profesyoneller tarafından uygulanırsa daha iyi olabileceğine dair görüş bildirilmiş özellikleri bir yöntemde birleştirerek, teorideki Scrum metodolojisi ile organizasyonların uyguladıkları Scrum modelinin farkları göstermeye imkan sağlamıştır. Çalışma kapsamında farklı organizasyonlarda uygulanan farklı özellikleri ve henüz uygulanamayan yeni pratikleri öneren bu model, yapılan değerlendirme sonuçları doğrultusunda güncellenerek iyileştirilmiştir. Böylece Scrum'ın ana felsefesine aykırı olacak, alan yazında Scrum ama olarak nitelendirilen ve bazı araştırmacılar tarafından da

istenmeyen bir durum olarak belirtilen uygulamaların organizasyonlarda kontrolsüz rastgele yapılmasının önüne geçilmesi hedeflenmektedir.

8.4. Çalışmanın Kısıtları ve Gelecek Çalışmalar

Çalışma içerisinde, anket ve görüşmeler ile ulaşılabilen profesyoneller genel olarak Türkiye'de bulunan organizasyonlarda ve projelerde görev aldıklarından dolayı, sonuçlar dünyada yer alan tüm organizasyonlara genellenemez. Önerilen Harmanlanmış Scrum modelinin organizasyonlarda uygulanmamış olması ve anket katılımcılarının genel olarak yazılım sektöründe görev alan katılımcılardan oluşmasına özen gösterilmesine rağmen, bu katılımcıların organizasyonlarında kullandıkları yazılım türlerine yer verilmemiş olması da çalışma içerisindeki kısıtlar arasında yer almaktadır. Özellikle yapılan görüşmelerin yalnızca 2 organizasyon ve sınırlı sayıda profesyonel ile gerçekleştirilmesi ve bu görüşmelerden elde edilen geri bildirimlerin önerilen model üzerinde kullanılması, çalışma sonunda değerlendirmeye katılan Scrum profesyonellerinin Harmanlanmış Scrum modeli içerisinde yer alan günlük Scrum toplantılarının kaldırılması ile ilgili olumsuz görüş belirtmelerinde etken olarak görülmektedir.

Gelecek çalışmalar kapsamında ise, Scrum yazılım geliştirme yöntemi ile ilgili deneyimleri olan veya bu konu ile ilgili çalışan daha fazla katılımcıya ulaşarak Harmanlanmış Scrum Modeli ile ilgili bulgular sağlamlaştırılıp çeşitlendirilebilir. Ek olarak bu modelin birebir uygulandığı projeler sağlanmalı ve bu projelerden alınacak geribildirimler değerlendirilmelidir.

KAYNAKLAR

Abrahamsson P., Haikara J., Pikkarainen M., Salo O., Still J., The Impact of Agile Practices on Communication in Software Development, *Empirical Software Engineering*, 2008, **13**(3), 303-337.

Abrahamsson P., Salo O., Agile Methods in European Embedded Software Development Organisations: a Survey on the Actual Use and Usefulness of Extreme Programming and Scrum, *Institution of Engineering and Technology Software*, 2008, **2**(1), 58-64.

Abrahamsson P., Ronkainen J., Salo O., Warsta J., Agile Software Development Methods–Review and Analysis, Vtt Publications, http://www.vtt.fi/inf/pdf/publications/2002/P478.pdf, (Ziyaret tarihi: 07 Mayıs 2016).

Akbayır D., Bayraklı S., Çamoğlu K., Yücalar F., Bir Çevik Yazılım Geliştirme Sürecinin Uyarlanması ve Uygulanması, *Havacılık ve Uzay Teknolojileri Dergisi*, 2010, **4**(3), 57-67.

Akdoğan E. N., Uçar T. Z., Middle School Students' Understanding of Average, Elemantary Education Online, http://ilkogretim-online.org.tr/vol8say2/v8s2m10.pdf, (Ziyaret tarihi: 05 Mart 2016).

Alliance A., Manifesto for Agile Software Development, Agile manifesto, http://www.agilemanifesto.org, (Ziyaret tarihi: 01 Mart 2016).

Agile Türkiye, Software Productivity Report, Ambysoft, http://www.agileturkey.org/, (Ziyaret tarihi: 01 Şubat 2015).

Ambler S., Agile Adoption Rate Survey Results: February 2008, Ambysoft, http://www.ambysoft.com/downloads/surveys/AgileAdoption2008.pdf, (Ziyaret tarihi: 01 Ocak 2015).

Ambler S., Agile Project Success Rates Survey Results, Ambysoft, http://www.ambysoft.com/downloads/surveys/AgileSuccess2010.pdf, (Ziyaret tarihi: 02 Ocak 2015).

Ambler S., Agile Adoption Mini-Survey, Ambysoft, http://www.ambysoft.com/downloads/surveys/AgileAdoption2014.pdf, (Ziyaret tarihi: 01 Ocak 2015).

Attanasio F., Scrum Mastering Reloaded, Google, https://books.google.com.tr/books?id=2vTjCgAAQBAJ&printsec=frontcover&hl=tr&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false, (Ziyaret tarihi: 01 Şubat 2016).

Azizyan G., Kajko-Matsson M., Magarian M. K., Survey of Agile Tool Usage and Needs, *In Agile Conference*, 2011, DOI: 101109/AGILE.2011.30.

Balçiçek Ö.E., Agile Proje Yönetimi, Slideshare, http://www.slideshare.net/okkesemin/agile-proje-ynetimi-10625113, (Ziyaret tarihi: 01 Mayıs 2016).

Başar A., Özkaya A., Kesgin F., Yazılım Geliştirme Süreçlerinde Şelale Yönteminden Çevik Yaklaşıma Geçiş: Bir Teknoloji Şirketinde Uygulama, *9. Ulusal Yazılım Mühendisliği Sempozyumu – UYMS,* Yaşar Üniversitesi, İzmir, 9-11 Eylül 2015.

Baytam V., Kalıpsız O., Scrum Yazılım Geliştirme Metodolojisi için Yönetim Sistemi Tasarımı ve Gerçeklenmesi, Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul, 2011, 297062.

Baytam V., Kalıpsız O., Scrum Yazılım Geliştirme Modeli Yönetim Aracı: ScrumMApp, *5. Ulusal Yazılım Mühendisliği Sempozyumu – UYMS*, ODTÜ Kültür Kongre Merkezi, Ankara, 26-28 Eylül 2011.

Bazeley P., Issues in Mixing Qualitative and Quantitative Approaches to Research, *Applying Qualitative Methods to Marketing Management Research*, 2002, **21**(2), 91-98.

Beck K., Beedle M., Van Bennekum A., Cockburn A., Cunningham W., Fowler M., ... , Kern J., Manifesto for Agile Software Development, Agile Manifesto, http://domingogallardo.github.io/apuntes-mads/s02-valores-y-principios-agiles/s02_4-agile-manifesto.pdf (Ziyaret tarihi: 05 Haziran 2016).

Beedle M., Schwaber K., *Agile Software Development with Scrum*, Pearson International Edition, Prentice Hall, New York, 2001.

Beedle M., Schwaber K., *Agile Software Development with Scrum*, Pearson International Edition, Upper Saddle River, Prentice Hall, New York, 2002.

Black J. A., Champion D. J., *Methods and Issues in Social Research*, John Wiley and Sons, New York, 1976.

Boehm B. W., A Spiral Model of Software Development and Enhancement, *The Institute of Electrical and Electronics Engineers Computer,* DOI: 10.1109/2.59.

Bryman A., Quantitative and Qualitative Research: Further Reflections on Their Integration, Editors: Brannen J., *Mixing Methods: Qualitative and Quantitative Research*, Avebury, Aldershot, London, 57-78, 1992.

Bucanac C., Johansson C., The V-Model, IDE, University of Karlskrona, Ronneby, http://bucanac.com/documents/The_V-Model.pdf, (Ziyaret tarihi: 01 Mayıs 2015).

Bustard D., Greer D., Wilkie G., The Maturation of Agile Software Development Principles and Practice: Observations on Successive Industrial Studies in 2010 and 2012, *Engineering of Computer Based Systems*, DOI: 10.1109/ECBS.2013.11.

Chapman R., Coll R. K., Choices of Methodology for Cooperative Education Researchers, *Asia-Pacific Journal of Cooperative Education*, 2000, **1**(1), 1-8.

Cho J., A Hybrid Software Development Method for Large-scale Projects: Rational Unified Process with Scrum, *Journal of Issues in Information Systems*, 2009, **5**(2), 340-348.

Cohen L., Manion L., Morrison K., *Research Methods in Education*, 7th ed., Routledge, New York, 2011.

Cohen D., Lindvall M., Costa P., An Introduction to Agile Methods, *Advances in Computers*, DOI: 10.1016/S0065-2458(03)62001-2.

Çetin E., Çevik Yaklaşımlar Anket Çalışması, Surveey, https://docs.google.com/file/d/0B6Uc3xMvXFRkTlJWb2lQM2VZeDQ/edit (Ziyaret tarihi: 01 Mayıs 2015).

Denzin N. K., Lincoln Y. S., Introduction: Entering the Field of Qualitative Research, Editors: Denzin N. K., Lincoln Y. S., *Handbook of Qualitative Research*, 2nd ed., Thousand Oaks, Sage Publications, California, 6461-6466, 1994.

Dingsoyr T., Dyba T., Empirical Studies of Agile Software Development: A Systematic Review, *Information and Software Technology*, 2008, **50**(9), 833-859.

Dominguez, J. The Curious Case of the Chaos Report 2009, Project Smart, http://www.projectsmart.co.uk/the-curious-case-of-the-chaos-report-2009.html (Ziyaret tarihi: 10 Haziran 2016).

Durasiewicz S., Lassenius C., Paasivaara M., Using Scrum in Distributed Agile Development: A Multiple Case Study, *Global Software Engineering*, DOI: 10.1109/ICGSE.2009.27.

Eloranta V. P., Koskimies K., Mikkonen T., Exploring ScrumBut—An Empirical Study of Scrum Anti-patterns, *Information and Software Technology*, DOI: 10.1109/APSEC.2013.72.

Gall D. M., Borg R. W., Gall P. J., *Educational Research: An Introduction*, 6th ed., Longman, New York, 1996.

Galorath D., Software Project Failure Costs Billions-Better Estimation and Planning Can Help, Filed Under Project Management, http://www.galorath.com/wp/software-project-failure-costs-billions-better-estimation-planning-can-help.php, (Ziyaret tarihi: 16 Eylül 2014).

Glesne C., Peshkin, A., *Becoming Qualitative Researchers: An Introduction*, 5th ed., Longman, London, 1992.

Guerra-Garc C., Ju, R., Huertas C., Jim S., Problems in the Adoption of Agile-Scrum Methodologies: A Systematic Literature Review, *4th International Conference in Software Engineering Research and Innovation*, DOI: 10.1109/CONISOFT.2016.30.

Hayata T., Han J., A Hybrid Model for IT Project with Scrum, *Service Operations, Logistics and Informatics (SOLI) 2011 IEEE International Conference*, Beijing, 10-12 Haziran 2011.

Highsmith J. A., *Agile Software Development Ecosystems*, Addison-Wesley Longman Publishing Professional, Boston, MA, USA, 2002.

Hope B., Huff S. L., Link S., Strode D. E., Coordination in Co-located Agile Software Development Projects, *Journal of Systems and Software*, 2012, **85**(6), 1222-1238.

Hundermark P., Do Better Scrum, Scrumsense, http://www.scrumsense.com/wp-content/uploads/2009/12/DoBetterScrum-v2.pdf, (Ziyaret tarihi: 20 Ocak 2014).

Husen T., Research Paradigms in Education, Slideshare, http://www.slideshare.net/suchetanapawar/research-paradigms-in-education, (Ziyaret tarihi: 10 Mart 2015).

Jakobsen C. R., Johnson K., Sutherland J., Scrum and CMMI Level 5: The Magic Potion for Code Warriors, *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, Waikoloa, HI, 7-10 Ocak 2008.

Janoff N. S., Rising L., The Scrum Software Development Process for Small Teams, *The Institute of Electrical and Electronics Engineers Computer Software*, 2000, **17**(4), 26-32.

Jeffries R., Fractional Scrum or Scrum-But, Agile Atlas, http://agileatlas.org/articles/item/fractional-scrum-or-scrum-but, (Ziyaret tarihi: 25 Mayıs 2016).

Karasar N., *Bilimsel Arastirma Yöntemi*, 28th ed., Nobel Yayin Dagitim, Ankara, 2005.

Kazak N., Olay İncelemesinin Temel Adımları, Sosyal Bilimlerde Araştırma Yöntemleri, http://80.251.40.59/education.ankara.edu.tr/aksoy/eay/eay/b0506/ymetin.doc, (Ziyaret tarihi: 25 Mayıs 2016).

Kır G., Özaydın Ö., Scrum Adoption in Kuveyt Turk IT, Yüksek Lisans Tezi, T.C. Doğuş Üniversitesi, Bilim ve Teknoloji Enstitüsü, İstanbul, 2014, 376050.

Krishna V., Basu A., Scrum+:: Is it ScrumBut or ScrumAnd, *2011 Annual IEEE India Conference*, Hyderabad, 16-18 Aralık 2011.

Labuschagne L., Marnewick C., Factors that Influence the Outcome of Information Technology Projects in South Africa: An Empirical Investigation, *Acta Commercii*, 2009, **9**(1), 78-89.

Larman C., *Agile and Iterative Development: a Manager's Guide*, 2nd ed., Addison-Wesley Professional, Boston, 2004.

Livermore J., Factors that Impact Implementing an Agile Software Development Methodology, *Proceedings 2007 IEEE Southeastcon*, DOI: 10.1109/SECON.2007.342860.

Lundqvist K., Srinivasan J., Using Agile Methods in Software Product Development: a Case Study, *Information Technology: New Generations, 2009 ITNG'09 Sixth International Conference*, Las Vegas, 27-29 April 2009.

Mary S. A., Suganya G., Progression towards Agility: A Comprehensive Survey, *Computing Communication and Networking Technologies (ICCCNT) 2010 International Conference*, Karur, 29-31 July 2010.

Marnewick C., Labuschagne L., Factors that Influence the Outcome of Information Technology Projects in South Africa: An Empirical Investigation, *Acta Commercii*, 2009, **9**(1), 78-89.

Oppenheim A. N., Questionnaire Design, Interviewing and Attitude Measurement, http://www.fao.org/docrep/w3241e/w3241e05.htm, (Ziyaret tarihi: 05 Mayıs 2015).

Özdemir M., Nitel Veri Analizi: Sosyal Bilimlerde Yöntembilim Sorunsalı Üzerine Bir Çalışma, *Osmangazi Üniversitesi Sosyal Bilimler Dergisi*, 2010, **11**(1), 323-343.

Özgüven İ. E., *Araştırma, Seçmede Psikolojik Danışmada Görüşme İlke ve Teknikleri*, İleri Matbaası, Türkiye, 1980.

Patton Q. M., *Qualitative Evaluation and Research Methods*, 2nd ed., Sage Publication, London, 1990.

Pressman R. S., *Software Engineering: a Practitioner's Approach*, 8th ed., McGraw-Hill Education, Palgrave Macmillan, 2005.

Preuss D. H., Digital Focus Unveils Market Survey Results at Agile 2006, Digital Focus, http://www.infoq.com/news/Digital-Focus-Unveil-Survey-2006, (Ziyaret tarihi: 30 Mayıs 2014).

Pries-Heje J., Pries-Heje L., Why Scrum Works: a Case Study from an Agile Distributed Project in Denmark and India, *Agile Conference 2011*, Salt Lake City, 7-13 Agust 2011.

Pshigoda G., Smits H., Implementing Scrum in a Distributed Software Development Organization, *Agile Conference 2007*, Washington, 13-17 Agust 2007.

Royce W. W., Managing the Development of Large Software Systems, *Proceedings of the Institute of Electrical and Electronics Engineers Computer*, 1970, **26**(8), 1-9.

Sandelowski M., Focus on Research Methods-whatever Happened to Qualitative Description?, *Research in Nursing and Health*, 2000, **23**(4), 334-340.

Schindler C., Agile Software Development Methods and Practices in Austrian IT-Industry: Results of an Empirical Study, *Computational Intelligence for Modelling Control and Automation, 2008 International Conference*, Vienna, 10-12 Aralık 2008.

Schwaber K., Scrum Development Process, Scrum Papers, https://www.scruminc.com/scrum-papers/, (Ziyaret tarihi: 05 Ocak 2016).

Schwaber K., Scrum Development Process, Chalmers, http://www.cse.chalmers.se/~feldt/courses/agile/schwaber_1995_scrum_dev_process.pdf, (Ziyaret tarihi: 05 Ocak 2016).

Schwaber K., Sutherland, J., Business Object Design and Implementation, *OOPSLA*, 1995, Texas, 15-19 Ekim 1995.

Schwaber K., Agile Project Management with Scrum, Microsoft Press, http://dbmanagement.info/Books/MIX/Agile_Project_Management_With_Scrum.pdf (Ziyaret tarihi: 21 Şubat 2016).

Schwaber K., Sutherland, J., The Scrum Papers: Nuts, Bolts and Origins of an Agile Process, Scrum Papers, http://www.crisp.se/scrum/books/ScrumPapers20070424.pdf, (Ziyaret tarihi: 03 Eylül 2015).

Schwaber K., Sutherland J., Scrum Guide, Scrum, http://www.scrum.org/storage/scrumguides/Scrum_Guide.pdf (Ziyaret tarihi: 10 Ekim 2015).

Schwaber K., Sutherland J., What is Scrum, Scrum Alliance, http://www.scrumalliance.org/system/resource/file/275/whatIsScrum.pdf, (Ziyaret tarihi: 01 Haziran 2014).

Schwaber K., ScrumButs and Modifying Scrum, Scrum, http://www.scrum.org/scrumbut, (Ziyaret tarihi: 20 Mayıs 2016)

Seidman I., *Interviewing as Qualitative Research: a Guide for Researchers in Education and the Social Sciences*, 3rd ed., Teachers College Press, New York, 2013.

Sommerville I., *Software Engineering*, 9th ed., Addison-Wesley, England, 2000.

Sutherland J., Agile Development: Lessons Learned from the First Scrum, Scrum Alliance, http://www.scrumalliance.org/resources/35, (Ziyaret tarihi: 14 Ekim 2014).

Sutherland J., ScrumBut Test Aka the Nokia Test, Jeff Sutherland, http://jeffsutherland.com/scrumbutttest.pdf, (Ziyaret tarihi: 15 Mayıs 2016)..

Sutherland J., Nokia Test 20090325, Jeff Sutherland, http://jeffsutherland.com/nokiatest.pdf (Ziyaret tarihi: 20 Mayıs 2016).

Şimşek H., Yıldırım A., *Sosyal Bilimlerde Nitel Arastirma Yöntemleri*, 9th ed., Seçkin Yayıncılık, Ankara, 2013.

Sütçü C., Proje Yönetim Metodolojisi Nedir?, Word Press, https://cemsutcu.files.wordpress.com/2010/10/7-8-3-proje-yonetim-metodolojisi-nedir.pdf, (Ziyaret tarihi: 30 Aralık 2015).

Tarhan A., Yılmaz S. G., Çevik Süreç ile Artırımsal Sürecin Nicel Karşılaştırması: Bir Durum Çalışması, *5. Ulusal Yazılım Mühendisliği Sempozyumu – UYMS*, ODTÜ Kültür Kongre Merkezi, Ankara, 26-28 Eylül 2011.

Tsui F. F., Karam O., Bernal B., *Essentials of Software Engineering,* 3rd ed., Jones and Bartlett Publishers, America, 2013.

Türnüklü A., Eğitimbilim Araştırmalarında Etkin Olarak Kullanılabilecek Nitel Bir Araştırma Tekniği: Görüşme, *Kuram ve Uygulamada Eğitim Yönetimi*, 2000, **6**(4), 543-559.

URL-1: http://www.scrumforteamsystem.co.uk/, (Ziyaret tarihi: 05 Şubat 2016).

URL-2: https://www.scrum.org/scrumbut, (Ziyaret tarihi: 25 Mayıs 2016).

URL-3: http://www.versione.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf, (Ziyaret tarihi: 01 Haziran 2014)

URL-4: http://stateofagile.versionone.com/ 2016, (Ziyaret tarihi: 01 Haziran 2016)

URL-5: https://en.wikipedia.org/wiki/V-Model (Ziyaret tarihi: 10 Mayıs 2016).

Välimäki A., Kääriäinen J., Patterns for Distributed Scrum—a Case Study, *Enterprise Interoperability III,* DOI: 10.1007/978-1-84800-221-0_7.

Vijayasarathy L. E. O. R., Turk D., Agile Software Development: A Survey of Early Adopters, *Journal of Information Technology Management*, 2008, **19**(2), 1-8.

Warma R., The Success of Agile Software Development, Fontys University of Applied Sciences, http://docplayer.net/2824572-The-success-of-agile-software-development.html, (Ziyaret tarihi: 10 Mayıs 2014).

Wasson C.S., System Analysis, Design And Development: Concepts, Principles and Practices, *Wiley Online Library,* DOI: 10.1002/0471728241.

Wolcott H. F., Transforming Qualitative Data: Description, Analysis and Interpretation, Sage Publishing, https://uk.sagepub.com/en-gb/eur/transforming-qualitative-data/book4328, (Ziyaret tarihi: 10 Mayıs 2014).

Yin R.K., *Case Study Research Design and Methods,* 5th ed., Sage Publication, California, 2002.

Yitmen M., Diyetteyim ama, Agile Turkey, http://www.agileturkey.org/#!diyetteyim-ama/c1ru8, (Ziyaret tarihi: 25 Mayıs 2016).

EKLER

EK-A

A.1 ANKET SORULARI

A.1.1 Anket Bilgi Formu

Sayın Katılımcı,

Bu anket formu Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Ana Bilim Dalı'nda Yrd. Doç. Dr. Pınar Onay Durdu danışmanlığında Esra Çetin tarafından hazırlanan "Yazılım Geliştirme Organizasyonları için Harmanlanmış Scrum Çerçevesi" başlıklı yüksek lisans tezi kapsamında yapılan araştırma için hazırlanmıştır.

Anketteki ifadeler "Çevik (agile) yazılım geliştirme metodolojileri hakkındaki düşüncelerinizi ve görüşlerinizi saptamaya" yönelik bilimsel bir araştırma ile ilgilidir. Bu araştırmadan elde edilecek sonuçlar bilimsel ahlaka uygun olarak ve gizlilik içerisinde değerlendirilecektir. Değerli zamanınızı ayırdığınız için teşekkür ederiz.

Araştırma sonuçlarının sizinle de paylaşılmasını isterseniz e-posta adresinizi belirtmeniz yeterli olacaktır. Eğer anketle ilgili ya da araştırmamızla ilgili soru, eleştiri ve önerileriniz olursa aşağıdaki iletişim bilgileriyle bize ulaşabilirsiniz.

Yrd. Doç. Dr. Pınar Onay Durdu
Kocaeli Üniversitesi Bilgisayar Mühendisliği Bölümü

pinar.onaydurdu@kocaeli.edu.tr

Esra Çetin
Kocaeli Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği

esrasolguncetin@gmail.com

A.1.2 Genel Anket Soruları

Tablo A.1.2.

1- Yaşınız?					
2- Cinsiyetiniz?	Erkek		Kadın		
3- En son mezun olduğunuz derece ?	Lise	Yüksek Okul	Üniversite	Yüksek Lisans	Doktora
4- Toplam iş deneyiminizi kaç yıldır?					
5- Çalıştığınız kurumun hizmet verdiği sektör nedir?	Finans, Sigorta, Telekominikasyon, Bilgi Teknolojileri, Sağlık, Devlet, Diğer				
6- Kaç yıldır yazılım/BT sektöründe yer almaktasınız?					
7- Yazılım süreci içerisindeki göreviniz nedir?	Takım Üyesi, Proje Lideri, BT Müdürü, Yazılımcı, Operasyonel Destek, Geliştirici, İş analisti, Testçi ,Ürün Sahibi, Diğer				
8- Çalıştığınız organizasyonun toplam personel sayısı nedir?					
9- Şu anda bulunduğunuz kurumda ne kadar süredir görev yapmaktasınız?					
10- Kurumunuzda kullanılan yazılım süreç modeli aşağıdaki yaklaşımlardan hangisine karşılık gelmektedir?	Çağlayan, Evrimsel, Spiral, Artırımlı, V model, Scrum, XP, RUP, Diğer				

A.1.3 Çevik Yaklaşım Anket Soruları

Tablo A.1.3.

11- Kurumunuzda ne kadar süredir çevik yaklaşımlar ile projeler yürütülmektedir?	<1	1-2 yıl	3-4 yıl	5-10 yıl	10+	
12- Kurumunuzda çevik yaklaşımları kullandığınız proje sayısı nedir?	1-5	6-10	11-20	21-50	51+	
13- Kurumunuzda çevik yaklaşımlar ile çalışan en büyük takımın büyüklüğü nedir?	1-5	6-10	11-20	21-50	51+	
14- Çevik projelerde genellikle ne kadar tekrarlama(yinelem e) yapılmaktadır?	1 hafta	2 hafta	3 hafta	4 hafta	5+ hafta	
15- Çevik projelerde kurumunuzda ortalama kaç kişilik proje takımları oluşturulmaktadır.	1-5	6-10	11-20	21-50	51+	
16- Kurumunuzda gerçekleştirilen çevik projelerin başarı oranı nedir?	%10'dan az	%10-%25	%26-%50	%51-%75	%76-%100	
17- Kurumunuzda çevik yazılım süreç yönetimi için kullandığınız özel bir araç var mı?	Evet		Hayır			
18- Kurumunuzda kullanılan çevik yazılım süreç yönetimi için hangisine karşılık gelmektedir?	TFS (Team Foundatio n Server) - Takım ortak bilgi yönetim sunucusu	MSF (Microsoft Solutions Framework) - MSF Çevik ve MSF CMMI	enVision® - Doküman ve Süreç Yönetim Sistemi	SmartAge Singularity - Süreç Yönetim Platformu	ENSEMB LE - Süreç Yönetim Sistemi	Diğer

A.1.4 Çevik Yaklaşım Değerlendirme Soruları

Aşağıda yazılan ifadeleri çalıştığınız kurumdaki çevik yaklaşımların kullanımı açısından **Kesinlikle Katılıyorum** ve **Kesinlikle Katılmıyorum** arası bir değerle değerlendirmenizi rica ederiz.

Tablo A.1.4.

	Kesinlikle Katılmıyorum	Katılmıyorum	Kararsızım	Katılıyorum	Kesinlikle Katılıyorum
19. Çevik yaklaşımları kullanmak verimliliğimizi arttırdı.	1	2	3	4	5
20. Çevik yaklaşımlar üretilen sistemlerinin kalitesini arttırdı.	1	2	3	4	5
21. Çevik yaklaşımlar geliştirme maliyetini azalttı.	1	2	3	4	5
22. Çevik yaklaşımlar paydaşların memnuniyetini arttırdı.	1	2	3	4	5
23. Çevik takımlar yeterli dokümantasyon hazırlamaz.	1	2	3	4	5
24. Çevik takımlar yeterli mimari tasarım yapmaz.	1	2	3	4	5
25. Çevik takımlar yeterli analiz yapmaz.	1	2	3	4	5
26. Çevik takımlar yeterli planlama yapmaz.	1	2	3	4	5
27. Çevik yazılım geliştirme disiplinsizdir.	1	2	3	4	5
28. Çevik yazılım geliştirme sadece geçici bir hevestir.	1	2	3	4	5
29. Çevik yazılım geliştirme sadece bir arada çalışan takımlara uygundur.	1	2	3	4	5

	Kesinlikle Katılmıyorum	Katılmıyorum	Kararsızım	Katılıyorum	Kesinlikle Katılıyorum
30. Çevik yazılım geliştirme sadece küçük takımlar içindir.	1	2	3	4	5
31. Çevik yazılım geliştirmenin sonucu düşük kalitedir.	1	2	3	4	5
32. Çevik yazılım geliştirme projelerini yönetmek zordur.	1	2	3	4	5
33. Eklemek istediğiniz görüşleriniz varsa, lütfen belirtebilir misiniz?					

A.1.5 Geleneksel Süreç Modeli Soruları

Tablo A.1.5.

34. Seçtiğiniz yazılım sürecinin işinizi kolaylaştıran özellikleri nelerdir?	Kolay kullanılabilirlik	Esneklik	Takip edilebilirlik	Öğretici bilgiler	Diğer
35. Kullandığınız yazılım sürecinin bir özelliğini çıkarmak, uygulamamak isteseydiniz bu hangisi olurdu?					
36. Kullandığınız yazılım sürecine bir özellik eklemek isteseydiniz bu ne olurdu?					
37. Organizasyonunuzun ne kadar sürede çevik yazılım geliştirme tekniklerine uyum sağlayacağınız düşünüyorsunuz?	3 ay	3-6 ay	6-12 ay	12-24 ay	Bilmiyorum
38. Bir proje yönetim aracı geliştiriyor olsaydınız nelere dikkat ederdiniz?	Kolay kullanılabilirlik	Esneklik	Takip edilebilirlik	Öğretici bilgiler	Diğer
39. Ek görüşleriniz varsa belirtebilir misiniz?					

EK-B

B.1 GÖRÜŞME SORULARI

B.1.1 Katılımcılar, Organizasyonları ve Projeler ile İlgili Bilgiler

1. **Kısaca kendinizi tanıtabilir misiniz?**
    a. Kurum içerisindeki göreviniz (ünvan) / sorumluluklarınız?
    b. Ne kadar süredir bu kurumda görev alıyorsunuz?
    c. Bulunduğunuz birim?

2. **Şirket bilgileriniz nelerdir?**
    a. Organizasyonunuzun büyüklüğü- çalışan sayınız nedir?
    b. Kurumunuz ne kadar zamandır faaliyet gösteriyor?
    c. Hangi sektörlere yönelik projeler geliştiriyorsunuz?
        i. Organizasyonlarınızda geliştirilen proje türleri nelerdir?

3. **Projelerinizi geliştirirken çevik yazılım geliştirme yöntemlerini kullanıyor musunuz?**

4. **Çevik yazılım geliştirme ile ilgili herhangi bir eğitime sahip personeliniz var mı?**
    a. İlgili personelin ünvanı nedir?
    b. Eğitim bilgisi nedir?
        i. Diploma programı
        ii. Sertifika programı
        iii. İlgili bir ders alma
        iv. İş başında kazanılan deneyim
    c. İlgili ünvana sahip personel yoksa bu sorumluluğu kim nasıl gerçekleştiriyor?

5. **Kurumunuzda çevik yazılım geliştirme ile ilgili eğitim veriyor musunuz?**
    a. Eğitim bilginiz?
        i. Diploma programı
        ii. Sertifika programı
        iii. İlgili bir ders alma
        iv. İş başında kazanılan deneyim

B.1.2 Organizasyon içerisinde çevik yazılım geliştirme kavramı ve uygulaması

6. **Kurumunuzda uygulanan çevik yazılım geliştirme metodolojisini tanımlayabilir misiniz?**

7. **Kurumunuzda ne kadar süredir çevik yaklaşımlar ile projeler yürütülmektedir?**

8. **Kurumunuzda çevik yaklaşımları kullandığınız proje sayısı nedir?**

9. **Çevik projelerde kurumunuzda ortalama kaç kişilik proje takımları oluşturulmaktadır?**

10. **Kurumunuzda çevik yaklaşımlar ile çalışan en büyük takımın büyüklüğü nedir?**

11. **Projelerinizde belirteceğim konular için öncelik sırası nedir?**

    i. Verimlilik
    ii. Kalite
    iii. Maliyet
    iv. Müşteri memnuniyeti

12. **Kurumunuzda hangi çevik yazılım geliştirme metodolojilerini kullanıyorsunuz?**

    a. Scrum proje yönetiminin kurumunuzdaki genel algısı nedir?
    b. Belirteceğim özelliklerden hangileri Scrum proje yönetiminde öne çıkmaktadır?

        i. Kolay kullanılabilirlik
        ii. Esneklik
        iii. Takip edilebilirlik
        iv. Öğretici bilgileri

B.1.3 Kurumunuzda Scrum Geçişi ve Rolleri

13. **Kurum olarak Scrum'a geçiş süreciniz ne kadar zaman aldı?**

    a. Scrum'a geçişte kurumunuzu zorlayan özellikler nelerdir?
    b. Scrum öncesi ve Scrum sonrası olarak ayırdığımızda kurumunuzda ne gibi değişiklikler fark ettiniz?

14. **Scrum içerisinde yer alan belirteceğim rollerden tümünün kurumunuzda karşılığı var mıdır?**

    i. Ürün Sahibi
    ii. Takım
    iii. Scrum Master

15. **Scrum projesi içerisindeki görev tanımınız hangisine karşılık gelmektedir?**

    i. Ürün Sahibi
    ii. Takım Üyesi
    iii. Scrum Master

16. **Ürün sahibinin kurumunuz içerisinde proje yönetimine katkı açısından rolü nedir?**

17. **Kurumunuz içerisinde Scrum master için tanımlanan proje yönetimine katkı rolü nedir?**

18. **Kullandığınız çevik yazılım geliştirme metodolojisinin kurumunuza has uygulanma farklılıkları var ise açıklar mısınız?**

    a. **Sizi bu şekilde farklı uygulamaya yönlendiren nedir?**

19. **Belirteceğim Scrum toplantılarından hangileri kurumunuzda yapılmaktadır?**

    i. Sprint Planlama Toplantısı
    ii. Günlük Scrum Toplantısı (Dün ne yaptım, yarın ne yapacağım ve engelim nedir, 15 dk)
    iii. Sprint Değerlendirme Toplantısı (Sprint sonunda, hedefe ulaştık mı,1 aylık sprint için max 4 saat)
    iv. Retrospektif- Geçmişe Bakış (Sprint değerlendirme ve yeni Sprint planlama toplantısı arasında yer alır)

20. **Belirteceğim Scrum terimlerinden hangileri kurumunuzda uygulanmaktadır?**

    i. Ürün İş Listesi (Product Backlog)
    ii. Sprint İş Listesi (Sprint Backlog)
    iii. Kalan Zaman Grafikleri (Burndown-Charts)
    iv. Engel İçeriği (Impediment Backlog)
    v. Bitti Tanımı (Definition of Done)

21. **Kurumunuzda geliştirilen projelerde çevik yazılım geliştirme metodolojisine uygunluk konusuna ne derece önem verilmektedir?**

    a. Sizce kurum içerisinde bu konuda yeterli düzeyde farkındalık var mıdır?

B.1.4 Çevik Yazılım Geliştirmede Araç Kullanımı

22. **Kurumunuzda çevik yazılım süreç yönetimi için kullandığınız özel bir araç var mıdır?**

a. Kullandığınız araçtan memnun musunuz?
b. Kullandığınız aracın genel özellikleri nelerdir?
c. Bu proje yönetim aracına bir özellik eklemek isterseniz ne eklerdiniz?
d. Bu proje yönetim aracından bir özellik çıkarmak isterseniz ne çıkarırdınız?
e. Kullanmıyorsanız; bu yönde bir ihtiyaç hissediyor musunuz?

B.1.5 Organizasyonlarda Çevik Yazılım Geliştirme Proje Yönetimi ve Başarı Durumu

23. Çevik yazılım geliştirme metodolojilerinin proje yönetimine katkısı nedir?

a. Çevik yazılım geliştirme içerisinde uygulamak isteyip de uygulayamadığınız aşamalar var mı?

24. Kurumunuzda gerçekleştirilen çevik projelerin başarı oranı nedir?

25. Başarısız olduğunuz durumlar var ise belirteceğim özelliklerden hangisi veya hangileri neden olarak gösterilebilir?

i. Dokümantasyon yetersizliği
ii. Mimari tasarım yetersizliği
iii. Analiz yetersizliği
iv. Planlama yetersizliği
v. Bir arada çalışamama durumu
vi. Gereğinden fazla büyük takım çalışması
vii. Proje yönetim zorluğu

B.1.6 Çevik Yazılım Geliştirme ile İlgili Profesyonellerinin Algıları

26. Çevik yazılım geliştirme metotlarının proje yönetimine sağladığı avantaj ve dezavantajlar sizce nelerdir?

27. Çevik yazılım geliştirme metotlarının proje yönetimine ne derece etki edebildiğini düşünüyorsunuz?

a. Proje yönetiminde çevik yazılım geliştirme metotlarının daha etkili veya değiştirilmeden uygulanabilmesi için gerekli faktörler nelerdir?

28. Çevik yazılım geliştirme metodolojilerinin projelerde etkili bir biçimde uygulanamama nedenleri sizce neler olabilir?

29. Şimdiye kadar geliştirdiğiniz projelerden edindiğiniz tecrübeler doğrultusunda çevik yazılım geliştirme metotları ile ilgili bize verebileceğiniz öneriler nelerdir?

EK-C

C.1 GÖRÜŞMELERİN ANALİZİ

C.1.1 Katılımcı ve Kategori Kodları

Tablo C.1.1.

Kategori no	Kategori adı	Katılımcı kodu	Cevaplar
KN_1_1	Çevik yöntem ile ilgili eğitimli personel	O1_SM	Var. Scrum ile ilgili organizasyon bünyesinde kendimizi geliştiriyoruz.
		O1_TU	Var.
		O2_SM	Var. Scrum ile ilgili yapılan PSM (Professional Scrum Master) sınavına giriyoruz. Ben bir puanla kaybettim.
KN_1_2	Çevik yöntemin kullanım süresi	O1_SM	Ben 1 senedir kurumdayım. Geldiğimden beri Scrum yazılım geliştirme yöntemi kullanılıyor.
		O1_TU	Birkaç yıldır Scrum yazılım geliştirme yöntemlerini kullanıyoruz.
		O2_SM	3 yılı aşkın bir süredir Scrum yazılım geliştirme yöntemini kullanıyoruz.
KN_1_3	Çevik yöntemin kullanıldığı proje sayısı	O1_SM	Bu bankada çevik proje olarak yaptığım 3. proje, hatta 4. Kurumdaki proje sayısı ise çok fazladır.
		O1_TU	Çevik projeleri bankada uzun süredir çok fazla projede kullanıyoruz, ben şimdiye kadar 7 veya 8 tane çevik yazılım kullanılan projede görev aldım.
		O2_SM	Tek bir projemiz var, aynı proje için sürekli yeni sürüm çıkarıyoruz, sadece çevik yöntem kullanıyoruz, 9 yıldır aynı projede görev alıyorum.
KN_1_4	Çevik yöntem kullanan proje takımının büyüklüğü	O1_SM	Her Scrum takımı kapalı alanda bir odada beraber çalışıyor. İdeal olan bu, genelde 7,8 kişi değişiklik gösteriyor. 4,5 kişide var 12,13 kişi de var.
		O1_TU	Şimdiye kadarki en büyük takımımız 13 kişilikti.
		O2_SM	İki Scrum takımı içerisinde Scrum master olarak görev almaktayım. Biri 8 diğeri 7 kişilik olan bu takımlar içerisinde bir ürün sahibi, bir Scrum Master, birer ekip lideri ve ayrıca Scrum takım üyeleri bulunmaktadır. Şimdiye kadar çalışan en büyük Scrum takımı ise 15 kişiliktir. İki takım bir arada tek bir odada çalışmaktadır.

KN_1_5	Çevik yöntemin proje yönetimine sağladığı avantaj ve dezavantajlar	O1_SM	Scrum'ın genel mantığı kaynakları tek odada çalıştırma fakat, çoğu zaman, kaynakları sürekli aynı odada tutamıyoruz. Hiyerarşide proje bazlı organizasyon olmadığı için, projedeki her kaynak farklı bölüm müdürlükleri altında yer alıyor ve başka iş öncelikleri oluyor. Tek bir proje ile baştan sona kadar ilgilenemiyorlar.
		O1_TU	Projelerimizde Scrum'ın avantajlar sağladığını söyleyebiliriz. Örneğin, Scrum'ın planlama sürecinde esneklik sağladığını gözlemliyoruz.
		O2_SM	Takip edilebilirliğin Scrum yazılım geliştirme yönteminin öne çıkan özelliği olduğunu söyleyebiliriz. Analiz, dokümantasyon ve planlamaları ise yeterlidir. Scrum yöntemine geçtikten sonra iş yükü azalmış işler daha ölçülebilir hale gelmiştir.
KN_2_1	Organizasyonda yer alan Scrum rolleri	O1_SM	Bizim kullandığımız Scrum mantığında bir proje lideri var. Benim kimliğim burada Scrum master ve Proje lideri. Çevik projelerde proje lideri isterse etkin rol alabilir.
		O1_TU	Belli arkadaşlar belli görevler üstleniyorlar, scrum master, analist, testçi, geliştirici, ürün sahibi gibi…
		O2_SM	Scrum Master, ekip lideri, ürün sahibi ve Scrum takım üyeleri bulunmaktadır.
KN_2_4	Scrum pratikleri kullanımı	O1_SM	Günlük Scrum toplantısı, Sprint planlama toplantısı, Sprint değerlendirmesi yapılıyor.
		O1_TU	Scrum pratiklerini uyguluyoruz, sadece geçmişe bakış teorik Scrum'da bulunan ancak pratikte yer almayan bir uygulama, Sprint değerlendirmenin sonunda yapıyoruz diyebiliriz.
		O2_SM	Günlük Scrum toplantısı yapmıyoruz; Haftalık Scrum toplantısı yapıyoruz, ayrıca Sprint planlama toplantısı, Sprint değerlendirmesi ve geçmişe bakış yapıyoruz.
KN_2_3	Organizasyonlarda Scrum Toplantıları Süreleri	O1_SM	Günlük Scrum toplantılarımız 15 dakikayı geçmemektedir. Sprint planlama toplantıları 2-3 saat aralığındadır.
		O1_TU	Toplam Sprint süremiz 3 haftadır. Sprint değerlendirme toplantılarında ise 4 saati geçirmemekteyiz.

		O2_SM	Günlük Scrum toplantılarını yapmamaktayız ancak, 1 saatlik haftalık Scrum toplantıları yapmaktayız. Sprint planlama toplantılarının süresi 1-2 saat olarak değişmektedir. Toplam Sprint süremiz 7 haftadır. Sprint değerlendirme toplantılarımız 1-2 saat aralığında değişmektedir.
KN_2_3	Scrum kavramları kullanımı	O1_SM	Ürün iş listesi, kullanıcı hikayeleri, iş bitim grafikleri, bitti tanımı kavramlarını kullanıyoruz.
		O1_TU	Beyaz tahta ve engel içeriği haricindeki Scrum kavramlarını kullanıyoruz.
		O2_SM	Tüm Scrum kavramlarını kullanıyoruz.
KN_3_1	Geleneksel yöntemlerden Scrum yöntemine geçiş süresi	O1_SM	Çevik yöntem dışında bankada şelale yöntemi de kullanılıyor. Hatta çok daha fazla kullanılıyordu, ekiplere göre değişiklik gösteriyor. Bazı takımlar çevik takım olarak oluşturuldu. Onların her yaptıkları projede çevik yöntem kullanılıyor, ancak bankadaki bazı takımlarda hala şelale yöntemini de kullanıyor.
		O1_TU	Çevik yöntemlere geçişimiz birkaç yıl öncesine dayanıyor, ancak iki yöntemi de hala kullanıyoruz geçiş süreci için bir yıllık uyum süresi harcadığımızı söyleyebiliriz.
		O2_SM	Bir yıl.
KN_3_2	Geleneksel yöntemlerden Scrum yöntemine geçişte zorlanılan durumlar	O1_SM	Ben organizasyona dahil olduğumdan beri çevik yöntemleri kullanıyoruz, geçiş sürecinde yer almadım.
		O1_TU	Scrum yazılım geliştirme yöntemine geçiş esnasında büyük zorluklar yaşanmadı, Scrum pratiklerine ve kavramlarına kolaylıkla uyum sağlandı.
		O2_SM	Personelin Scrum pratikleri, kavramları ve araçları ile ilgili eğitim süreci dışında herhangi bir olumsuz durum yaşanmadı, uyum sağlayamadığımız Scrum özellikleri için teorideki Scrum yöntemi özellikleri dışında, kendi organizasyonumuza uygun özellikler ekledik.
KN_4_1	Proje yönetim aracı kullanımı	O1_SM	TFS (Team Foundation Server - Takım ortak bilgi yönetim sunucusu) proje yönetim aracını kullanıyoruz.
		O1_TU	Kullanıyoruz. TFS.
		O2_SM	Java platformu üzerinde geliştirilmiş, Jira proje yönetim aracını kullanıyoruz.
KN_4_2	Proje yönetim aracının genel özellikleri ve	O1_SM	TFS proje yönetim aracından memnunuz.
		O1_TU	TFS, Scrum için uygun bir proje yönetim aracıdır.

	memnuniyet algısı	O2_SM	Jira proje yönetim aracı her türlü raporu kolayca alabileceğimiz şekilde tasarlanmıştır. Araçtan genel olarak memnunuz.
KN_4_3	Proje yönetim aracına eklenmesi gereken özellik	O1_SM	Ürün iş listesindeki madde güncellemelerinin kolay yapılabilmesi adına iş akışlarında güncellemelerin hızlandırılması.
		O1_TU	TFS proje yönetim aracının daha esnek bir yapıya sahip olması için gerekli düzenlemelerin yapılması
		O2_SM	Jira'nın çağrı merkezi ile entegrasyon özelliğinin eklenmesi
KN_4_4	Proje yönetim aracından çıkarılması gereken özellik	O1_SM	İş akışlarının hızlandırıcı ve esnek yapıda değil, statü güncellemeleri zor yapılıyor.
		O1_TU	Raporlamadaki yer alan karmaşıklığın giderilerek esnekliğin sağlanması gerekiyor.
		O2_SM	Jira'nın gereksiz özellikleri bulunmamaktadır, bu yüzden çıkarılması gereken özellik yoktur.

EK-D

D.1 HARMANLANMIŞ SCRUM MODELİ DEĞERLENDİRME MADDELERİ

D.1.1 Harmanlanmış Scrum Modeli Değerlendirme Bilgi Formu

Sayın Katılımcı,

Bu değerlendirme formu Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Ana Bilim Dalı'nda Yrd. Doç. Dr. Pınar Onay Durdu danışmanlığında Esra Çetin tarafından hazırlanan "Yazılım Geliştirme Organizasyonları için Harmanlanmış Scrum Modeli" başlıklı yüksek lisans tez çalışması kapsamında yapılan araştırma için hazırlanmıştır.

Formda yer alan ifadeler tez çalışması kapsamında önerilmekte olan "Harmanlanmış Scrum Modeli hakkındaki düşüncelerinizi ve görüşlerinizi saptamaya" yönelik bilimsel bir araştırma ile ilgilidir. Harmanlanmış Scrum modeli, Scrum yazılım geliştirme yöntemi ile birçok ortak noktası olan, ancak farklı süreçleri, pratikleri ve özellikleri de içinde barındıran bir model olarak önerilmektedir. Görüşleriniz önerilen Harmanlanmış Scrum modelinin Türkiye'de Scrum yazılım geliştirme yöntemini uygulayan organizasyonlara uyumu ile ilgili bilgi edinmek için kullanılacaktır. Bu araştırmadan elde edilecek sonuçlar bilimsel ahlaka uygun olarak ve gizlilik içerisinde değerlendirilecektir. Değerli zamanınızı ayırdığınız için teşekkür ederiz.

Araştırma sonuçlarının sizinle de paylaşılmasını isterseniz e-posta adresinizi belirtmeniz yeterli olacaktır. Eğer anketle ilgili ya da araştırmamızla ilgili soru, eleştiri ve önerileriniz olursa aşağıdaki iletişim bilgileriyle bize ulaşabilirsiniz.

Esra Çetin
Kocaeli Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği
esrasolguncetin@gmail.com

Yrd. Doç. Dr. Pınar Onay Durdu
Kocaeli Üniversitesi Bilgisayar Mühendisliği Bölümü
pinar.onaydurdu@kocaeli.edu.tr

D.1.2 Demografik Bilgiler

Tablo D.1.2

1- Yaşınız?					
2- Cinsiyetiniz?	Erkek		Kadın		
3- En son mezun olduğunuz derece ?	Lise	Yüksek Okul	Üniversite	Yüksek Lisans	Doktora
4- Toplam kaç yıl iş deneyiminiz olduğunu belirtebilir misiniz?					
5- Çalıştığınız kurumun hizmet verdiği sektör nedir?	Finans, Sigorta, Telekomünikasyon, Bilgi Teknolojileri, Sağlık, Devlet, Diğer				
6- Kaç yıldır yazılım/BT sektöründe yer almaktasınız?					
7- Yazılım süreci içerisindeki göreviniz nedir?	Takım Üyesi, Proje Lideri, IT Müdürü, Yazılımcı, Operasyonel Destek, Geliştirici, İş analisti, Testçi ,Ürün Sahibi, Diğer				
8- Çalıştığınız organizasyonun toplam personel sayısı nedir?					
9- Şu anda bulunduğunuz kurumda ne kadar süredir görev yapmaktasınız?					
10- Kurumunuzda projeler Scrum süreci ile mi geliştirilmektedir?	Evet, Hayır				
11- Şu ana kadar görev aldığınız Scrum proje sayısını belirtebilir misiniz?					
12- Organizasyonunuzda kaç yıldır Scrum süreci kullanılmaktadır?					

D.1.3 Değerlendirme Soruları

Aşağıda Scrum takımı rol ve sorumlulukları, Scrum toplantıları, Scrum araçları ve Sprint süreci ile ilgili Harmanlanmış Scrum Modelinde yapılan tanımlamaların kullanımlarının uygunluğu açısından "**Kesinlikle Katılmıyorum"** ve "**Kesinlikle Katılıyorum"** arası 1'den 5'e kadar bir değerle değerlendirmenizi rica ederiz. Fikrinizin bulunmadığı maddeler için 0 numaralı **"Fikrim yok"** seçeneğini işaretleyebilirsiniz.

Fikrim yok: 0
Kesinlikle Katılmıyorum: 1
Katılmıyorum: 2
Kararsızım: 3
Katılıyorum: 4
Kesinlikle Katılıyorum: 5

Tablo D.1.3

SCRUM TAKIMI ROL VE SORUMLULUKLARI						
Scrum takımı genel özellikleri						
	Fikrim yok	Kesinlikle Katılmıyorum	Katılmıyorum	Kararsızım	Katılıyorum	Kesinlikle Katılıyorum
	0	**1**	**2**	**3**	**4**	**5**
1. Scrum takım üyeleri aynı oda içerisinde çalışmalıdır.						
2. Scrum takım üyeleri sürekli iletişim halinde olmalıdır.						
3. Scrum takım üyeleri, analiz, yazılım ve test safhalarının tümünde görev alabilen çalışanlardan oluşmalıdır.						
4. Scrum takımının asli görevi her sprint sonunda çalışan bir yazılımı ürün sahibine sunmaktır.						
Scrum takımı üyelerine *proje lideri rolü eklenmesi önerilmektedir*. Bu bağlamda proje liderinin sahip olması öngörülen rol ve sorumluluklar ile ilgili ifadeler hakkındaki görüşlerinizi belirtiniz.						
5. Proje lideri ürün sahibi veya Scrum master olmamalıdır.						
6. Proje lideri bir ekip değil, kişi olmalıdır.						
7. Proje lideri yetki tanımı, Scrum master'ın bazı sorumlulukları ile geleneksel yazılım geliştirme modellerindeki proje liderlerinin görevlerini içermelidir.						
8. Proje lideri üst yönetim ve Scrum takımı arasında gerekli koordinasyonu sağlamalıdır.						
9. Proje lideri ekip içerisinde kaynak (örn. personel) yönetimini sağlamalıdır.						
10. Proje lideri Scrum takım üyelerinin iş önceliklerini belirlemelidir.						
11. Proje lideri efor tahmini sırasında gerekli durumlarda müdahale etmelidir.						

	Fikrim yok	Kesinlikle Katılmıyorum	Katılmıyorum	Kararsızım	Katılıyorum	Kesinlikle Katılıyorum
	0	**1**	**2**	**3**	**4**	**5**
12. Proje lideri hangi modülün hangi takım üyesi tarafından yapıldığını bilmelidir.						
13. Proje lideri takım üyeleri arasında direk iş paylaşımı yapmamalı, takım içerisinde anlaşmazlık olduğu durumlarda müdahale etmelidir.						
14. Proje lideri takım üyeleri üzerinde baskı kurmamalıdır.						
15. Proje lideri yazılımcı, analistçi gibi takım üyelerinden biri olabilir.						
16. Scrum takım üyelerini proje lideri yönetmelidir.						
17. Proje lideri rolü ile Scrum takımı gerektiğinde yönetilebilen bir takım özelliği göstermelidir.						
18. Scrum takımı üyelerine yukarıdaki maddelerde belirtilen rol ve sorumluluklara sahip proje lideri rolü eklenmelidir.						
Scrum takımı içerisindeki diğer roller ve sorumluluklar ile ilgili ifadeler hakkındaki görüşlerinizi belirtiniz.						
19. Scrum master, Scrum sürecini yönetmeli ve Scrum pratiklerini kontrol etmelidir.						
20. Scrum master, Scrum takımının Scrum pratiklerine ve kavramlarına uygun kontrol etmelidir.						
21. Proje lideri ve Scrum master koordine bir şekilde çalışmalıdır.						
22. Ürün sahibi müşterinin isteklerini proje ekibine anlatan kişidir.						
23. Ürün sahibi, müşteri ile Scrum proje ekibi arasındaki etkileşimi sağlamalıdır.						
24. Ürün sahibi, ortaya çıkarılacak ürünü yönetmelidir.						
25. Ürün sahibi, ürün iş listesinde yer alacak gereksinimleri ve kullanıcı hikayelerini belirler.						
26. Organizasyon içerisinde yer alan faaliyetlerden etkilenen kişilerin veya sistemlerin oluşturduğu paydaşlar, gerekli durumlarda Scrum takımına dahil olmalıdır.						
27. Son ürünü kullanacak olan kullanıcılar gerekli durumlarda Scrum takımına dahil olmalıdır.						
28. Scrum takımı rol ve sorumlulukları ile ilgili eklemek istediğiniz görüşleriniz varsa, lütfen belirtebilir misiniz?						
SCRUM TOPLANTILARI						

Sprint Planlama Toplantısı						
	Fikrim yok	Kesinlikle Katılmıyorum	Katılmıyorum	Kararsızım	Katılıyorum	Kesinlikle Katılıyorum
	0	**1**	**2**	**3**	**4**	**5**
29. Sprint planlama toplantıları her sprint öncesinde yapılmalıdır.						
30. Sprint planlama toplantılarına, Scrum takım üyeleri, Scrum master, ürün sahibi ve proje lideri katılmalıdır.						
31. Sprint planlama toplantılarında, ürün iş listesinde yer alan işler içerisinden ilgili Sprint'e dahil olacak işlerin seçimi ve efor tahmini yapılmalıdır.						
Günlük ve Haftalık Scrum Toplantıları						
32. Günlük Scrum toplantıları yerine bir saat sürecek haftalık Scrum toplantılarının yapılması uygundur.						
33. Haftalık Scrum toplantılarında, "Geçen hafta ne yaptım? Bu hafta ne yapacağım? Karşılaşılan problemler ve yenilikler nelerdir?" soruları Scrum takım üyeleri tarafından cevaplanmalıdır.						
34. Haftanın belirlenen bir gününde, opsiyonel olarak yapılabilecek hafta arası Scrum toplantısı yapılması uygundur.						
35. Hafta arası Scrum toplantısı Scrum takım üyelerinin, Scrum master, ürün sahibi veya proje liderinin talebi ile gerçekleştirilebilir.						
36. Hafta arası Scrum toplantıları 15-20 dakikayı geçmemelidir						
37. Scrum toplantıları oturarak yapılmalıdır.						
38. 15 dakikalık günlük Scrum toplantıları yapılmamalıdır.						
Sprint Gözden Geçirme Toplantısı						
39. Sprint gözden geçirme toplantıları her sprint sonunda yapılmalıdır.						
40. Sprint gözden geçirme toplantısında ürünün sunumu yapılmalıdır.						
41. Geçmişe bakış toplantıları, ayrıca yapılmayarak Sprint gözden geçirme toplantılarının içerisine dahil edilmelidir.						
42. Scrum toplantıları ile ilgili eklemek istediğiniz görüşleriniz varsa, lütfen belirtebilir misiniz?						
SCRUM ARAÇLARI						

	Fikrim yok	Kesinlikle Katılmıyorum	Katılmıyorum	Kararsızım	Katılıyorum	Kesinlikle Katılıyorum
	0	**1**	**2**	**3**	**4**	**5**
43. Beyaz tahta Scrum toplantıları sırasında kullanılmalıdır.						
44. Scrum projesi içerisindeki tüm işler ürün iş listesine yazılmalıdır.						
45. Her sprint için sadece o sprint'e dahil olacak işler, sprint iş listesine alınmalıdır.						
46. Sprint iş listesinde yer alacak maddelerin ürün iş listesinde belirlenmiş olması gerekmektedir.						
47. Ürün iş listesine proje süresi boyunca yeni maddeler eklenebilmelidir.						
48. Proje sürecinde belirlenebilen engeller, Scrum master tarafından engel içeriği olarak kayıt edilmelidir.						
49. Kullanıcı hikayeleri müşteriden alınan gereksinimler ile oluşturulmalıdır.						
50. Bitti tanımı ile kullanıcı hikayelerinin tamamlanma kontrolü yapılmalıdır.						
51. Ürün ve sprint kalan zaman grafikleri, ilgili sprint içerisindeki iş takibini kolaylaştırmak için tek bir grafik üzerinde, sprint'e ait işleri gösterecek şekilde düzenlenmelidir.						
52. Scrum araçları ile ilgili eklemek istediğiniz görüşleriniz varsa, lütfen belirtebilir misiniz?						
SPRİNT SÜRECİ						
53. 4 haftalık Sprint süresi aşılmamalıdır.						
54. Sprint iş listesinde belirlenen maddelerin bir sonraki Sprint'e bırakılmamalıdır.						
55. Her Sprint kendi içerisinde analiz, tasarım, geliştirme, test ve teslim süreçlerini içermelidir.						
56. Sprint içerisinde hata çıkma olasılığı göz önünde bulundurularak, hata ayıklamaya makul bir zaman ayrılmalıdır.						
57. Sprint süresi sonuna opsiyonel olarak kullanılabilen maksimum bir hafta süreli, Sprint gözden geçirme zamanı eklenebilir.						
58. Sprint gözden geçirme zamanı Scrum takımı kararı doğrultusunda, 1 gün veya maksimum 7 güne kadar kullanabilmelidir.						

	Fikrim yok	Kesinlikle Katılmıyorum	Katılmıyorum	Kararsızım	Katılıyorum	Kesinlikle Katılıyorum
	0	**1**	**2**	**3**	**4**	**5**
59. Sprint gözden geçirme zamanı, Scrum takım üyelerine aynı Sprint içerisinde yer alan işleri gözden geçirme ve düzeltme imkanı sağlamalıdır.						
60. Sprint gözden geçirme zamanı, ekstra Sprint eklenmesini engelleyebilir.						
61. Sprint gözden geçirme zamanı, Sprint içerisinde teslim edilecek iş sayısının azaltıldığı durumlarda ek süre olarak kullanılabilir.						
62. Sprint süreci ile ilgili eklemek istediğiniz görüşleriniz varsa, lütfen belirtebilir misiniz?						

KİŞİSEL YAYINLAR VE ESERLER

[1] Durdu P. O., **Çetin E.**, Türkiye'de Çevik Yazılım Geliştirme Üzerine Bir İnceleme, *8. Ulusal Yazılım Mühendisliği Sempozyumu*, *UYMS*, ODTÜ, KKTC, 8-10 Eylül 2014.

ÖZGEÇMİŞ

1983 yılında İstanbul'da doğdu. İlk, orta ve lise öğrenimini İstanbul'da tamamladı. 2006 yılında Maltepe Üniversitesi Bilgisayar Mühendisliği bölümünden mezun oldu. 2012 yılında başladığı, Kocaeli Üniversitesi Bilgisayar Mühendisliği bölümündeki, yüksek lisans öğrenimine devam etmektedir.