SETSCI Conference Indexing System,
Volume 3 (2018), 1040-1045

Yazılım Proje Geliştirme Sürecinde Proje Yönetim Aşamaları ve Risk Analizi İncelemesi

Baki Gökgöz[1*+], Mustafa Keskinkılıç[2]

[1]*Bilgisayar Teknolojileri Bölümü, Torul Meslek Yüksekokulu, Gümüşhane Üniversitesi, Gümüşhane, Türkiye*
[2]*Atatürk Üniversitesi, İktisadi İdari Bilimler Fakültesi, Yönetim Bilişim Sistemleri Bölümü, Erzurum, Türkiye*
[*]*Corresponding author: bakigkgz@gmail.com*
[+]*Speaker: bakigkgz@gmail.com*
Presentation/Paper Type: Oral / Full Paper

Özet-Bu çalışmanın amacı, yazılım projelerinin geliştirilmesinde ve geliştirilen projeden en iyi verimin elde edilmesine büyük katkılar sağlayan, yazılım projesi geliştirme süreçlerindeki risk yönetiminin incelenmesidir. Ayrıca Yazılım proje geliştirme aşamaları adım adım incelenmiş ve bu adımlar yazılım proje yönetimi perspektifinde ifade edilmiştir. Günümüzde yazılım proje yönetimi çalışmaları büyük ölçüde pratiğe dökülememiş olması nedeniyle bu projelerdeki risk oranlarının azaltılamadığı saptanmıştır. Bu durum yazılım firmalarına zaman kaybı, maliyet ve başarısız projeler olarak negatif geri dönüşümler meydana getirmiştir. Yazılım proje geliştirme süreçlerinde bir risk analiz uzmanının istihdam edilmesinin projenin başarısı için çok önemli bir faktör olduğu saptanmıştır. Çalışmanın sonucunda tespit edilen bulgular ve öneriler paylaşılmıştır.

Anahtar Kelimeler- *Yazılım proje yönetimi, SDLC, yazılım riskleri, yazılım projelerinde risk, risk yönetimi*

Software Project Development Process Project Management Processes and Risk Analysis Review

Abstract- The purpose of this study is to examine risk management risk management in software project development processes, which provides significant contributions to the development of software projects and the best productivity from the project.In addition, software project development phases are reviewed step by step and these steps are expressed in software project management perspective.It has been determined that the risk ratios in these projects can not be reduced because software project management studies have not been widely practiced today.This has created negative returns for software companies as waste of time, cost and unsuccessful projects.It is also stated that the employment of a risk analysis expert in software project development processes is a very important factor for the success of the project.Findings and suggestions identified in the result of the study are presented in a list.

Keywords- *Software project management, SDLC, software risks, software project risk, risk management*

1. Giriş

Günümüzde yazılım teknolojisi tüm dünyada özellikle gelişmiş ülkelerde çok hızlı gelişen sektörlerin başında gelmektedir [1]. Yazılım teknolojisinin, maliyetin ve insan ilişkilerinin bir yazılım projesi bünyesinde iyi bir şekilde entegrasyonunun sağlanması oldukça zor bir iştir. Çok uzun bir süreci içeren yazılım projeleri, farklı niteliklerdeki insan gruplarının çalışmasına ve performansına dayanan yüksek yoğunluklu bir çalışmadır.Yazılım proje yönetimi geliştirilecek yazılım projelerinin kapsam, maliyet ve zaman üçgeninde en iyi verime sahip projenin geliştirilmesini amaçlamaktadır.

Günümüze kadar bir çok kişi proje yönetimini tanımlamaya çalışmıştır. Örnek olarak, Olsen, 1950'lerde en erken tanımlamalardan birini yapmıştır. Proje Yönetimi, farklı kaynakların kullanımını zaman, maliyet ve kalite kısıtlamaları dahilinde özgün, karmaşık, tek zamanlı bir görevin başarılmasına yönlendirmek için bir takım araç ve tekniklerin bir uygulamasıdır [2]. Başka bir tanıma göre proje yönetimi, yapılacak projenin amacına ulaşabilmesi için proje ile ilgili bütün adımların planlanması, programlanması ve kontrol işlemlerine tabi tutulması olarak tanımlanabilir [3]. Olsen tarafından kullanılan açıklamaya dahil edilen başarı ölçütleri, bugün proje yönetimini tanımlamak için kullanılmaya devam edilmektedir. İngiltere Proje Yönetimi Derneği (Association of Project Management) tarafından oluşturulan Birleşik Krallık Bilgi Birimi [4], proje yönetimi için bir tanım oluşturmuştur bu tanıma göre: Bir projenin tüm yönlerinin planlanması, organizasyonu, izlenmesi ve kontrol edilmesi ve projenin hedeflerine ulaşılması için gereken zaman, maliyet ve performans kriterleri dahilinde, güvenli ve öngörülen hedeflerin tümüdür. Proje yöneticisi bunu başarmak için tek sorumluluk noktasıdır.

2. Yazılım Proje Yönetimi ve Proje Yöneticisi

Yazılım proje yönetimi, yazılım ürünlerinin geliştirilmesini yönetmek için gerekli olan bilgi, teknik ve araçları kapsar. Günümüzdeki geniş kapsamlı yazılım projeleri düşünüldüğünde bu projeler birbiriyle bağlantılı bir çok modülden oluşmaktadır. Bunun için bu projelerin gelişim aşamalarında değişiklikler yapmak hem teknik açıdan hemde zaman ve maliyet açısından oldukça problemli bir durumdur. Birçok yazılım projesi hatalı proje yönetimi uygulamaları nedeniyle başarısız olmaktadır. Amerika da Standish Group International şirketinin 2004 yılında yapmış olduğu bir araştırmada geliştirilen projelerin %53'ünün gecikmiş veya proje için ayrılan bütçeyi aşmış, %18'inin proje analizleri ve risk faktörleri iyi bir şekilde yapılamadığı için bitirilememiş, veya değiştirilmiş olduğu belirlenmiştir. Yazılım projelerin %29'u ise başarılı bir şekilde sonlandırılmıştır. Bu araştırmanın ortaya koyduğu sonuç ise yazılım projelerinde geliştirme aşamasına geçmeden önce projenin iyi planlanması ve bu plan dahilinde yönetilmesi başarı olasılığını artırmak için bir anahtar rolü oynamaktadır [5].

Yazılım proje yöneticisinin öncelikli problemi, kullanıcılar, müşteriler, geliştirme ekibi, bakım ekibi ve yönetimden oluşan çeşitli destek grublarını bir yazılım projesinde eş zamanlı olarak organize etme geriğidir. Genellikle kullanıcılar çok fazla istekte bulunurlar, ayrıca geliştirilen projenin arzu ettikleri her şeyi yapmasını isteyebilirler. Müşteriler genellikle maliyete önem verirler yani düşük bütçeyle iyi bir ürün almak isterler. Proje yöneticisinin patronları ise normal olarak iddialı hedefleri olan iyi bir projeye sahip olmak istemektedirler. Ürünün koruyucuları, hatasız, iyi belgelenmiş, değiştirilmesi kolay bir sistem isterler. Geliştirme ekibi üyeleri ise genellikle geliştirme sürecinde meydana gelebilecek yönetimi zor olan teknik problemler ve zorluk durumları için projenin zaman kısıtının esnekleştirilmesini istemektedirler [6]. Bu istekler bir araya getirildiğinde temel çatışmalar meydana gelmektedir. Bu çatışma durumuna bir örnek: düşük bütçeyle çok fonksiyonel bir ürünün oluşturulmak istenmesidir. Bu çatışmalar, hem stratejik düzeyde hem de taktiksel düzeyde çoğu yazılım proje yönetimi zorluklarının temelidir [6]. İyi bir yazılım yönetimi teorisi, proje yöneticisinin bu zorlukları aşmasına yardımcı olmalıdır. Bir yazılım yönetim teorisi, istekleri yerine getirebilmek için benzer zorlu eş zamanlı hedeflere sahiptir. Projeyi anlamak ve bunu uygulamaya geçirmek basit olmalıdır. Peojeye ait tüm yapıları ve ilgili sınıfları kapsayacak kadar genel olmalıdır. Ayrıca proje yöneticisi proje için faydalı önerileri analiz edip sunacak kadar yeterli beceriye sahip olmalıdır. Projeye ait kaynakların optimum bir şekilde kullanılmasından ve bunlarla projeden

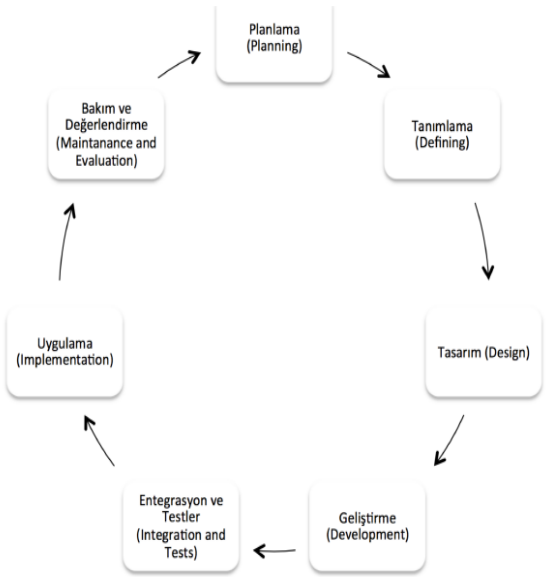
istenen verim ve sonucun alınmasından proje yöneticisi sorumludur.

3. Yazılım Geliştirme Sürecinin Temel Adımları

Her "Yazılım Projesi" nin bir proje olduğu gerçeği unutulmamalıdır [7]. Bunun için yazılımın belirtilen zamanda ve istenilen işlevlere sahip bir şekilde bitirilmesi çok önemlidir. Buda ancak iyi bir proje yönetimi sayesinde gerçekleşmektedir.

Yeni geliştirilen proje ile operasyonel süreç arasındaki dönem, Yazılım Geliştirme Yaşam Döngüsü veya kısaca SDLC (Software Development Life Cycle) olarak adlandırılmaktadır [8]. SDLC, bir yazılım organizasyonu içinde bir yazılım projesi için takip edilen süreçtir. SDLC, belirli yazılımların nasıl geliştirileceğinin, sürdürüleceğinin ve değiştirileceğinin açıklandığı ayrıntılı bir plandan oluşmaktadır. Yaşam döngüsü, yazılım kalitesi ve genel geliştirme süreçleri için bir metodoloji tanımlamaktadır. Yazılım geliştirme süreci uzundur ve bir takım tekniklere ihtiyaç duyar. SDLC, faydalı ve sağlam bir yazılım ürününün geliştirilmesini sağlamak için gerekli olan tüm aşamalardan oluşmaktadır. Ayrıca SDLC, maliyet etkin ve izlenebilir süreçleri de içermektedir.

Uygulama, web sitesi veya yazılımların geliştirilmesi karmaşık bir süreçtir. Bu geliştirme aşamalarının herhangi bir adımında yapılacak bir yanlış, hem ürün kalitesi hem de tüm iş için olumsuz sonuçlara neden olacaktır.



Şekil 1: SDLC Adımları [8].

3.1 Planlama

Planlama bir projeyi kusursuz bir şekilde başlatır ve ilerlemesini olumlu yönde etkiler. Planlama ve ihtiyaç analizi, SDLC'deki en önemli ve temel aşamadır. Müşterinin girdileri, satış departmanı, pazar anketleri ve sektördeki alan uzmanları ile ekibin kıdemli üyeleri tarafından gerçekleştirilir. Bu bilgi daha sonra temel proje yaklaşımını planlamak ve ekonomik, operasyonel ve teknik alanlarda ürün fizibilite çalışması yapmak için kullanılır. Kalite güvence gereksinimlerinin planlanması ve proje ile ilgili risklerin belirlenmesi de planlama aşamasında yapılır. Teknik fizibilite çalışmasının sonucu, projeyi minimum risklerle başarılı bir şekilde uygulamak için izlenebilecek çeşitli teknik yaklaşımları tanımlamaktır.

3.2 Tanımlama

Planlama ve ihtiyaç analizi yapıldıktan sonra, bir sonraki adım, ürün gereksinimlerini açık bir şekilde tanımlamak ve belgelemek ve bunları müşteriden veya pazar analistlerinden onaylatmaktır. Bu, proje yaşam döngüsü boyunca tasarlanacak ve geliştirilecek tüm ürün gereksinimlerinden oluşan bir Yazılım Gereksinim Dosyası (Software Requirements Specification (SRS)) belgesi ile yapılır.

3.3 Tasarım

Geliştirilecek projeye ait tasarımın yapıldığı aşamadır. Teknik olarak tasarım aşaması, veri, yazılım mimarisi, modüller ve arayüz tasarımı gibi farklı sınıflandırmalara ayrılabilir. Bununla birlikte çoğunlukla tasarım, modülerlik, veri yapıları ve yazılım mimarileri üzerinde yoğunlaşmaktadır [9]. Bu aşamada şekillerden faydalanılarak geliştirilecek projeye ait tasarım görselleştirilir. Bir önceki aşamada gerçekleştirilen planlama ve tanımlaya göre bir tasarım şeması çizilir. Yazılım Gereksinim Dosyası (SRS), ürün mimarlarının geliştirilecek ürün için en iyi mimari ile ortaya çıkması için bir referanstır. SRS'de belirtilen gerekliliklere dayanarak, genellikle ürün mimarisi için birden fazla tasarım yaklaşımı bir DDS(Tasarım Belgesi Spesifikasyonu) te önerilmiş ve belgelenmiştir. Bu DDS, tüm önemli paydaşlar tarafından gözden geçirilmekte ve risk değerlendirmesi, ürün sağlamlığı, tasarım modülerliği, bütçe ve zaman kısıtlamaları gibi çeşitli

parametrelere dayanarak, ürün için en iyi tasarım yaklaşımı seçilmektedir.

3.4 Geliştirme

SDLC'nin bu aşamasında gerçek gelişim başlar ve ürün inşa edilir. Bu aşamada programlama kodu DDS'ye göre oluşturulur. Tasarım ayrıntılı ve organize bir şekilde gerçekleştirilirse, kod üretimi çok güçlük çekmeden gerçekleştirilebilir.

Geliştiriciler, kuruluşlarının tanımladığı kodlama yönergelerini takip etmeli ve derleyicileri, yorumlayıcıları, hata ayıklayıcılarını vb. gibi programlama araçlarını kullanarak kodu üretmelidir. Kodlama için C, C ++, Pascal, Java ve PHP gibi farklı üst düzey programlama dilleri kullanılır. Programlama dili, geliştirilmekte olan yazılımın türüne göre seçilmektedir.

3.5 Entegrasyon ve Test

İyi planlanmış bir test süreci ile hatalar erken tespit edilerek zamandan, maliyetten ve enerjiden tasarruf edebilir. Testin kendisi önemli kaynakları tüketir ve kritik bir SDLC aşamasıdır. Test ekibi, enerji tüketimi ile ilgili iyi tanımlanmış test senaryoları oluşturmalı ve uygulamalıdır. Çeşitli koşullarda enerji tüketimi eğilimlerini değerlendirmek için güvenlik ve performans testinin sonuçlarını da analiz etmelidirler[9]. Geliştirilen sistemin artık gerçek hayata geçtiği aşamadır. Eğer bu sistem bir kurumda kullanılacak yazılım projesi ise kurumda bu yazılımı kullanacak personele yazılımla ilgili eğitimler verilmelidir. Ayrıca bu yazılım için kullanılacak donanımların tedarik edilmesi sağlanmalı ve bu donanımlar yazılıma entegre edilip test edilmelidir.

3.6 Uygulama

Bu aşamada geliştirilen proje ilgili kurumda kullanılır. Bu aşamada test aşmasında belirlenemeyen problemler ortaya çıkabilir. Ortaya çıkan bu problemlerle ilgili yeni çözümler üretilir. Yine bu aşamada yeni ihtiyaçlar ortaya çıkabilir. Bu problemler ve ihtiyaçlar proje geliştiricileri tarafından dikkate alınarak çözümler üretilir. Üretilen bu çözümler bir sonraki versiyon da projeye eklenir. Böylece daha kullanışlı ve verimli bir proje elde edilmiş olur.

3.7 Bakım ve Değerlendirme

Ürün test edildikten ve kullanılmaya hazır olduktan sonra, uygun pazarda resmen piyasaya sürülür. Bazen ürün dağıtımı, o organizasyonun iş stratejisine göre kademeli olarak gerçekleşir. Ürün ilk önce sınırlı bir segmentte piyasaya sunulabilir ve gerçek iş ortamında test edilebilir (UAT (User Acceptance Testing) - Kullanıcı Kabul Testi). Daha sonra, geri bildirime dayalı olarak, ürün olduğu gibi veya hedef pazar segmentinde önerilen geliştirmelerle piyasaya sunulabilir. Ürün piyasaya sürüldükten sonra, mevcut müşteri tabanı için bakım yapılır. Projenin bakımı yapılırken farklı aşamalardan geçirilir. Yeni tanımlar ortaya çıkar, yeni tasarımlar yapılır ve bu şeklinde devam eden döngü hiçbir zaman bitmez [8]. Sistem ne kadar iyi tasarlanırsa tasarlansın belirli bir yaşam süresi vardır. Bu süre sonunda başka sistemlere evrilir veya ölür. Sistemin yaşam döngüsünün uzun olması iyi tasarımına ve bakımına bağlıdır.

4. Yazılım Proje Yönetiminde Riskler

Risk ortaya çıkması durumunda projede istenen sonuca ulaşmayı engelleyecek olumsuz ve istenmeyen bir durumdur [10]. Yazılım projelerinde çok fazla bileşen olduğu için belirsizlik daha fazladır. Dolayısıyla bu sistemlerde risk ihtimali daha yüksek olmaktadır. Her yazılım projesinde risk tanımlaması ve yönetimi ana kaygılardandır. Bu durum geliştirilecek olan yazılım projelerinde risk yönetiminin çok önemli olduğunu göstermektedir. Yazılım risklerinin etkili analizi, etkili planlama ve iş atamalarına yardımcı olmaktadır.

Risk yönetimi beklenmeyen durumları ortadan kaldırmak veya en aza indirmek için yazılım geliştirme alanında en iyi metot olarak benimsenmektedir. Proje yöneticileri ileri düzeyde risk analizi becerilerine sahip olsalar da kesin geleceği bilemezler. Ancak iyi analizler yapılarak muhtemel riskler ve olumsuzluklar tahmin edilip bu durumların etkilerini en aza indirmek için çeşitli önlemler alınarak bu riskler yönetilebilmektedir. Proje yöneticisinin yaptığı bu iş risk yönetimi olarak adlandırılmaktadır. Diğer bir ifadeyle risk yönetimi, bir projeyle ilgili olası bir problemin projenin işleyişini aksatmadan önce bu problemin çözümü için alınan tedbirlerdir. Günümüzde risk yönetimi süreci çalışma alanları farklıda olsa da bütün bilimsel sistemlerde uygulanmaktadır. İstatistik, ekonomi, psikoloji, sosyal

bilimler, mühendislik, sistem analizi ve yöneylem araştırması alanları risk yönetimi süreçlerinden çok karşılaştığımız alanlardır. Kloman, çeşitli disiplinlerden oluşan bir dizi risk yönetimi biçimini akademik bir çalışmasında özetlemiştir. Örneğin risk yönetimi bankacılar için piyasa risklerine karşı para kaybının önlenmesi için alınan tedbirlerdir [10]. Geliştirilen bir sistemle ilgili olası risklerin tanımlanması risk yönetimi personeli tarafından çok dikkatli bir biçimde yapılması gerekmektedir. Geliştirilen sistemde görev alan herhangi bir çalışan tarafından bu risk analizleri ve bu risklere karşı alınacak önlemler değiştirilemez.

4.1 Yazılım Projelerinde Karşılaşılan Risk Türleri

Geliştirilen sistemlerin başarısız olma durumu yazılım projelerinde önemli bir problemdir. Riskler programın fiilen uygulanmasından önce tanımlanır, sınıflandırılır ve yönetilir. Bu riskler farklı kategorilerde sınıflandırılmaktadır. Literatür taramasında ilgili kaynaklarda yazılım proje geliştirme süreçlerinde karşılaşılan riskler genel olarak beş kategoride toplanabilir. Bu riskler; zaman, bütçe, operasyonel, teknik ve programatik risklerdir[11][12][13][14][15].

4.1.1 Zamanlama Riski

Proje görevleri ve yayım riskleri uygun şekilde ele alınmadığında proje takvimi kaybolabilmektedir. Bir projedeki zamanlama risklerine dikkat edilmezse bu durum projenin başarısız olmasına neden olabilir. Projenin başarısız olması ise şirket ekonomisinin olumsuz bir biçimde etkilenmesine yol açacaktır. Proje takviminin kaymasının genel nedenleri: yanlış zaman tahmini, kaynakların uygun şekilde incelenmemesi, beklenmeyen proje kapsam genişletmeleri olarak özetlenebilir.

4.1.2 Bütçe Riski

Bunlar: yanlış bütçe tahmini, maliyet aşımı, proje kapsam genişletmesi gibi bütçe aşımları ile ilişkili finansal risklerdir.

4.1.3 Operasyonel Riskler

Uygun olmayan süreç yönetimi uygulamalarından kaynaklanan riskler geliştirilen yazılım projesinin başarısızlığına neden olur. Operasyonel risklerin sebepleri: önceliklerin yanlış sıralanması, sorumlulukların yerine getirilmemesi, eksik kaynak, yeterli eğitimin olmaması (deneyimsizlik), kaynak planlamasının olmaması ve takımda iletişimin zayıf veya problemli olması.

4.1.4 Teknik Riskler

Teknik riskler genellikle işlevsellik ve performansın bozulmasına yol açar.

Teknik risklerin nedenleri genel olarak şunlardır: sürekli değişen ihtiyaçlar, ileri teknoloji mevcut değildir veya mevcut teknolojinin başlangıç aşamasında olması, ürünün uygulama için karmaşık olması ve proje modülleri entegrasyonun zor olması.

4.1.5 Programatik Riskler

Bunlar proje sınırlarının ötesinde olan dış risklerdir. Bu dış riskler şunlar olabilir: sermayenin azalması. müşterinin ürün stratejisini ve önceliğini değiştirmesi ve hükümetin yasal düzenlemeleri gibi dış riskler olabilir.

5. Risk Örnekleri

Bu konuda yapılan bilimsel araştırmalardan elde edilen sonuçlar, yazılım geliştirme süreçlerinde birçok riskin ortaya çıkabileceğini ifade etmektedir. Yapılan literatür taramasında birçok kaynak, muhtemel riskleri önlemek için önleyici tedbirler önermektedir. Tablo 2 de sık rastlanan risk türleri ve önlemleri gösterilmiştir [16].

Tablo 1
Boehm'in on risk faktörü

Risk faktörü	Önleyici tedbirler
1.Personel hatası	Personel işe alım sürecinde en iyi seçimi yapmak. Personelin ödüllendirmek. Takım Oluşumuna dikkat etmek.
2. Gerçekçi olmayan program ve bütçe	İş olurlukanalizi. Projeyi adım adım geliştirmek. Program ve bütçenin değiştirilmesi.
3. Standart yazılım, harici bileşenler (deneyimsizlik, uyumsuzluk vb.)	Kıyaslama yapmak. Prototip geliştirmek. Referans kurulumların gözden geçirilmesi. Uyumluluk analizi

	ve tedarikçilerin gözden geçirilmesi.
4. Gereksinimler ve gelişmiş fonksiyonların eşleşmemesi.	İlgili taraflar arasında kazan-kazan anlaşmaları. İş olurluk analizi. Prototip,erken aşamalarda uygulama tanımlaması.
5. Kullanıcı arayüzlerini ihtiyaçlara karşılamaması	Prototip senaryoların geliştirilmesi ve kullanıcıların tanımlanması.
6. Yetersiz mimari, performans ve kalite.	Simülasyon, kıyaslama, modelleme ve prototip.
7. Gereksinimlerin sürekli değiştirilmesi	Değişiklikler için artımlı eşik.Yönetim sürecini değiştirmek. Kontrol paneli değiştirmek.
8. Eski sistemlerle ilgili problemler.	Tasarım kurtarma ve yeniden yapılandırma
9. Dışfaktörler	Denetimler.Çeşitli tedarikçiler tarafından paralel tasarım veya prototipleme ve takım oluşumunu gerçekleştirmek.
10. Bilgisayar ve programlama yeteneklerinin zorlanması	Teknik Analiz, maliyet fayda analizi ve prototipleme işlemlerinin gerçekleştirilmesi.

6. Sonuç ve Öneriler

Bu çalışmada genel anlamda proje yönetimi incelenmiştir. Daha sonra çalışmada yazılım proje yönetimi ve yazılım proje yöneticisinin görevleri hakkında yapılan araştırmalar ve incelemeler paylaşılmıştır.

Bu çalışmada yazılım proje yönetiminin aşamaları olan SDLC çeşitli kaynaklardan araştırılmış ve bunlar paylaşılmıştır.

Bu çalışmanın amacı olan yazılım proje geliştirme de karşılaşılacak riskler incelenmiştir. Bu risklere karşı çözüm yolları araştırılmıştır.

Sonuç olarak başarılı bir yazılım projesi için iyi bir yazılım projesi yöneticisine ve olası riskleri en üst seviyede analiz edebilecek bir risk yöneticisine ihtiyaç olduğu ortaya konulmuştur. Böylece bir yazılım projesinde risk yönetimi araştırmaların da, olası riskler ne kadar erken teşhis edilip çözülürse, yazılım projeleri için ölümcül risk olan başarısızlık ihtimali önlenmiş olur. Ayrıca risklerin önceden isabetli bir şekilde tahmin edilip çözümlenmesi zaman ve maliyet kaybını da önleyecektir.

REFERANSLAR

[1] LLC Digital Publications. (2005). Software development lifecycle phases (V.1.1c , 2005).

[2] Oisen, R. P. (1971). *Can Project Management be Defined?*. Project Management Quarterly.

[3] Lewis P. J. (1996) . *Project Planing And Scheduling Control* (3rd Edition). Mc Grow Hill.

[4] BS 6079-1. (2010). British Standard in Project Management.

[5] Kwan, T. W., ve Leung, H. (2010). A Risk Management Methodology for Project Risk

[6] Boehm, B. W. ve Ross R. (1989). Theory-W Software Project Management: Principles and Examples (Vol. 15, no. 7). IEEE Transactıons on Software Engineering.

[7] Sherer, A. S. (1995). The three dimensions of software risk: technical, organizational, and *environmental.* Proceedings of the 28th Annual International Conference on System Sciences.

[8] Şeker, Ş. E. (2015). Yazılım Geliştirme Modelleri ve Sistem/Yazılım Yaşam Döngüsü. YBS Ansiklopedi. Cil t 2, Sayı 3, Eylül 2015

[9] Chauhan N. S. ve Saxena A. (2013). A Green Software Development Life Cycle for Cloud Computing. Infosys, India.

[10] Nizam A. (2015). *Yazılım Proje Yönetimi*. (2. Baskı). İstanbul: Papatya Yayınevi.

[11] Rivard S., James Y. S., ve Cameron, A. F. (2011). Software Project Risk Drivers as Project Manager Stressors and Coping Resources. Proceedings of the 44'th Hawaii International Conference on System Sciences. Hawaii, USA.

[12] Natarajan, K. V. (2004). Efficient Software Development. Proceedings of MASPLAS'04. Mid-Atlantic Student Workshop on Programming Languages and Systems. Seton Hall University. South Orange, New Jersey, U.S.

[13] Westfall, L. (2014). Software Risk Management. The Westfall Team http://westfallteam.com/Papers/risk_management_paper.pdf.

[14] Bodea, C., ve Dascalu, M. (2009). Modeling Research Project Risks with Fuzzy Maps. JAQM Journal of Applied Quantitative Methods, 4(2).

[15] Hoodat, H., ve Rashidi, H. (2009). Classification and Analysis of Risks in Software Engineering. World Academy of Science, Engineering and Technology.

[16] Calp M. H., ve Akcayol, M. A. (2015). Yazılım Projelerinde Karşılaşılan Risk Faktörleri ve
Risk Yönetim Süreci. Marmara Fen Bilimleri Dergisi, 1: 1-13.