Koc. Üni. Fen Bil. Der., 2(1): (2019) 36-42

Kocaeli Üniversitesi

Kocaeli Üniversitesi Fen Bilimleri Dergisi

http://dergipark.org.tr/koufbd

Yazılım Projelerinin Gözden Geçirme Sürecinde Hata Yoğunluğunu Etkileyen Faktörlerin Belirlenmesi: Bir Ar-Ge Kurumunda Uygulama Örneği

Determination of Factors Affecting the Defect Density in Review Process of Software Projects: A Case Study in a R&D Organization

Özgür GÜN[1*], Pınar Yıldız KUMRU[2]

[1]*Kocaeli Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği, Kocaeli, 41380, Türkiye,* ***Orcid Id:*** *0000-0002-4987-2980*
[2]*Kocaeli Üniversitesi, Mühendislik Fakültesi, Endüstri Mühendisliği, Kocaeli, 41380, Türkiye,* ***Orcid Id:*** *0000-0002-6729-7721*

Araştırma Makalesi

Gönderilme Tarihi : 11/02/2019
Kabul Tarihi : 09/04/2019

Anahtar Kelimeler

Bulgu Yoğunluğu
Deney Tasarımı
Gözden Geçirme Süreci

Özet

Yazılım projelerinde planlanan zamanda ve bütçede müşterinin istediği nitelikte ürün çıkarmak projenin ana hedefidir. Bir yazılım projesinin başlangıcından bitişine kadar proje planlama, analiz, tasarım, kodlama, tümleştirme, ve test süreçleri birbirleri ile etkileşimli olarak işletilir. Bu süreçlerin işletimi sonucunda ortaya çıkan ürünler (yazılım kodları, planlar, gereksinim dokümanları, tasarım dokümanları, test dokümanları, raporlar vb.) müşteriye teslim edilmeden önce gözden geçirme süreci ile gözden geçirilir. Müşteriye hatasız ürün vermeye destek olan bir süreç olduğu için gözden geçirme süreci bir projenin başarıyla bitirilmesinde önemli rol oynar. Bir gözden geçirme faaliyeti sonucunda hata sayısı, gözden geçirilen ürün büyüklüğü, gözden geçirme için harcanan süre, gözden geçirilen ürünlerin tür bilgileri elde edilir. Hata sayısının ürün büyüklüğüne oranı hata yoğunluğu bilgisini verir. Bu çalışmada yazılım projelerinde doküman gözden geçirmelerindeki hata yoğunluğuna etki eden faktörleri belirlemek için üç faktör, iki düzeyli ve dörder tekrarlı deney tasarımı gerçekleştirilmiştir. Hipotez testleri ile sonuçlar yorumlandığında; doküman gözden geçirmelerinde doküman büyüklüğü ve doküman tipi faktörlerinin hata yoğunluğuna anlamlı etkisinin olduğu, doküman gözden geçirme için harcanan işgücü faktörünün hata yoğunluğuna anlamlı etkisinin olmadığı görülmüştür.

Research Paper

Received Date : 11/02/2019
Accepted Date : 09/04/2019

Keywords

Defect Density
Design of Experiment
Review Process

Abstract

The main goal in software projects is to develop products that customer needs in the planned time and budget. From the beginning to the end of a software project planning, analysis, design, coding, integration, review and testing processes are implemented interactively with each other. The products resulting from the operation of these processes (software codes, plans, requirements documents, design documents, test documents, reports, etc.) are reviewed with the Review Process before being delivered to the customer. Review Process is very important for the successful completion of a project, as it is a process that supports the delivery of defect-free products to the customer. As a result of a review activity, we obtain the number of defects, the product size reviewed, the time spent for review, and the type information of the products reviewed. The ratio of the number of defects to the product size gives the result of the defect density. In this study three factors, two-level and four repetitive experiments were designed to determine the factors affecting the defect density in document reviews of software projects. When the results are interpreted by hypothesis testing; it was found that document size factor and document type factor had a significant effect on the defect density and the workforce factor spent on document review had no significant effect on the defect density.

[*]Sorumlu Yazar *(Corresponding Author):* ozgur.gun@tubitak.gov.tr

1. Giriş

Yazılım projelerinde planlanan zamanda ve bütçede müşterinin istediği nitelikte ürün çıkarmak projenin ana hedefidir. Müşteriye hatasız ürün vermeye destek olan bir süreç olduğu için gözden geçirme süreci bir projenin başarıyla bitirilmesinde önemli rol oynar. Bir yazılım projesinde üretilen ürünün tipine göre farklı gözden geçirme yöntemleri uygulanır. Bunlardan yönetimsel gözden geçirme proje planlarını, takvimini, kaynak durumlarını, kapsam değişikliklerini ve düzeltici faaliyetleri gözden geçirirken, teknik gözden geçirmeler ise kalifiyeli insanlar tarafından gerçekleştirilen ve projenin gereksinim, tasarım, test gibi teknik dokümanlarında yer alan bilgilerin amaçlanan kullanımına uygunluğunu değerlendiren faaliyetlerdir. Ayrıca denetim (audit), inceleme (inspection) ve eşli gözden geçirme (walk-throughs) tipleri de sistem ve yazılım ürünlerindeki hataları yakalamak, ürünü iyileştirmek, alternatif çözümleri göz önünde bulundurmak ve ürünün sözleşmede belirtilen standartlara uygunluğunu değerlendirmek amacıyla yapılan gözden geçirmelerdir [1].

Bir yazılım projesinin çıktılarının (yazılım kodları, proje planları, gereksinim dokümanları, tasarım dokümanları, test dokümanları, raporlar vb.) gözden geçirilmesi esnasında; müşteri ihtiyaçlarına, tasarım dokümanlarına, standartlara dayalı olarak beklentilerden sapma durumlarına "hata" adı verilir [2]. Birim ürün başına düşen hata sayısına "hata yoğunluğu" denir. Örneğin yazılım kodunda her bin kod satırı için raporlanan hata sayısı [3]. Bu çalışmada proje dokümanlarının proje ekibi tarafından gözden geçirilmesi faaliyetlerinden elde edilen hata yoğunlukları verileri kullanılmıştır.

Yazılım projelerinin planlama, analiz, tasarım, kodlama, tümleştirme ve test aşamalarında ortaya çıkan ürünler müşteriye teslim edilmeden önce gözden geçirilir. Bir gözden geçirme süreci aşağıdaki aktivitelerden oluşur [1]:

a) Gözden geçirmeye başlama kriterlerinin sağlanması,
b) Gözden geçirmeye girecek ürünlerin belirlenmesi,
c) Gözden geçirmeye katılacak kişilere duyuru yapılması,
d) Gözden geçirmede bulunan hataların kaydedilmesi (bireysel gözden geçirme),
e) Gözden geçirme toplantısının gerçekleştirilmesi,
f) Gözden geçirme toplantısında alınan kararlara göre ilgili ürünlerde düzeltmelerin yapılması,
g) Düzeltmesi yapılan ürünlerin kontrol edilmesi için daha önce aynı ürünleri gözden geçirenlere tekrar gönderilmesi,
h) Gözden geçirme yapanların tümünün onay vermesi ile gözden geçirmenin kapatılması.

Literatür incelendiğinde yazılım hata yoğunluğuna etki eden faktörlerin belirlenmesine yönelik çalışmalar yapıldığı görülmüştür.

Wedyan ve ark. ISBSG'den (The International Software Benchmarking Standards Group) almış oldukları proje verileri üzerinde yapmış oldukları amprik çalışmada regresyon ve ANOVA yöntemlerini kullanarak; hata yoğunluğu üzerinde proje büyüklüğü ve efor faktörlerinin negatif etkisinin olduğunu, proje ekibine verilen iş büyüklüğünün ise pozitif etkisinin olduğunu ileri sürmüşlerdir [4].

Zhang ve ark. yazılım geliştirme süreç metrikleri ile yazılım hata yoğunluğu arasındaki korelasyonu araştırmışlardır. Sonuçlar, yazılım metrikleri ile hatalar arasındaki korelasyonun zayıf olduğunu buna karşılık metrikler ile hata yoğunluğu arasında korelasyonun anlamlı olduğunu göstermiştir. Hata yoğunluğuna en çok etki eden metrikleri bulmuşlar ve bu metriklerin kalite iyileştirme ve hata yoğunluğu tespit etmede kullanılabileceğini ileri sürmüşlerdir [5].

Bosu ve ark. kod gözden geçirmenin etkinliğini etkileyen faktörleri araştırmıştır. Microsoft firmasında gerçekleştirilen projelerde yapılan gözden geçirme sonuçları incelemiş ve sonuç olarak; a) kurumda uzun yıllar çalışan, tecrübeli yazılımcıların daha etkin gözden geçirme yaptığını, b) gözden geçirilecek yazılım bileşenlerindeki satır sayısının artmasıyla gözden geçirmenin etkinliğinin düştüğünü, c) gözden geçirme etkinliğinin hesaplanmasıyla yazılım sistemindeki hangi dosya tiplerinde ya da hangi yazılım bileşenlerinde zayıflık olduğunun tespit edilebileceğini saptamıştır [6].

Mandhan ve ark. lineer regresyon yöntemini kullanarak yazılım kod metriklerinin (coupling, cohesion, weighted methods, depth, response, comment, line of code) hata yoğunluğa pozitif etkisinin olduğunu ileri sürmüşlerdir. Kullanılan verilerin sınıf seviyesinde olduğunu, yapılan çalışmanın metot seviyesine genişletilerek aynı sonuçların alıp alınmayacağının araştırılması gerektiğini belirtmişlerdir [7].

Jamimi, yazılım metriklerini kullanan bulanık mantık tabanlı yazılım hata tahmini yapan bir model önermiştir. Geliştirilen model gerçek yazılım projelerinde denenmiş ve tahmin doğruluğunun istenen düzeyde olduğunu belirtmiştir [8].

Shah ve ark. hata yoğunluğu türlerinden (standart hata yoğunluğu, ayrışık hata yoğunluğu) hangisinin yazılım projelerinin kalitesi göstergesi olarak daha uygun olduğunu araştırmışlardır. Sonuç olarak her iki türün de farklı kalite trendlerinin olduğunu belirtmişlerdir. Standart hata yoğunluğunun proje boyunca genel kalite trendini gösterirken, ayrışık hata yoğunluğunun lokal kalite trendini (örneğin bir sürüme ait) gösterdiğini ileri sürmüşlerdir [9].

Chen ve ark. yapay sinir ağları tabanlı yazılım hata tahmini yapan bir model geliştirmişlerdir. Bu model farklı büyüklükteki veri setlerini öğrenme verisi olarak alıp homojen ve gerçeğe yakın veriye dönüştürmektedir. Sonuç olarak modelin dönüştürdüğü gerçeğe yakın verilerin yazılım hata tahminini iyileştirdiği görülmüştür [10].

Xiao ve ark. havacılık endüstrisinde 2007-2014 yılları arasında geliştirilen 60 projenin test sonuçları üzerinde yaptıkları çalışmada uzmanlık bilgisinin ve tecrübenin fazla olduğu projeler ile uzmanlık bilgisinin az olduğu yeni projeleri karşılaştırmışlardır. Sonuç olarak uzmanlık ve tecrübenin fazla olduğu projelerde kritik seviye hata trendinin düştüğünü fakat düşük seviye hata trendinin değişmediğini, buna karşılık yeni projelerde ortaya çıkan kritik hata trendinin şiddetle arttığını tespit etmişlerdir [11].

Yamashita ve ark. yazılım büyüklük ve karmaşıklık değerlerinin yazılım hata yoğunluğuna etkisinin olup olmadığını araştırmışlardır. Yaptıkları çalışma sonucu; a) yazılım kod büyüklüğü ile hata yoğunluğu arasında bir ilişki olduğunu fakat buna karşılık kod karmaşıklığı ile hata yoğunluğu arasında aynı seviyede bir ilişki olmadığını, b) büyük yazılım dosyalarını daha küçük dosyalara ya da daha az karmaşık dosyalara bölmenin hata yoğunluğunu düşürmediğini aksine verimliliğe ters etki yaptığını belirtmişlerdir. Bu nedenle yazılım mühendislerinin yazılım büyüklük ve karmaşıklık eşik değerlerini seçerken dikkatli olmalarını tavsiye etmişlerdir [12].

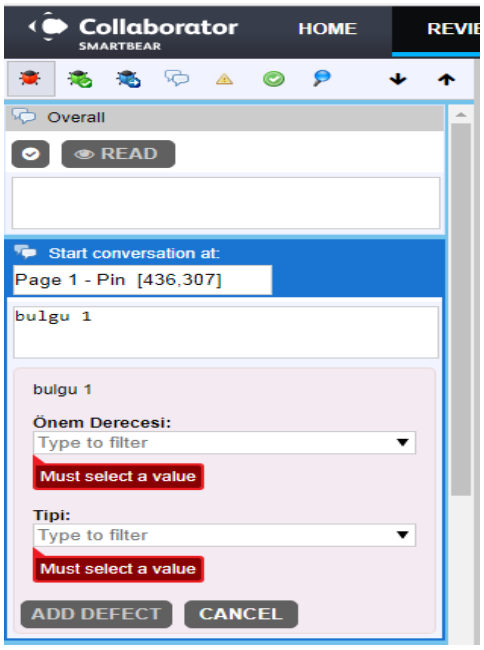
Alfadel ve ark. Halstead Complexity, Cyclomatic Complexity yazılım metriklerinin yazılım hata yoğunluğu ile güçlü ilişkisi olduğunu ileri sürmüşlerdir [13].

Hata yoğunluğuna etki eden faktörlerin belirlenmesinde çok etkenli deney tasarımı yönteminin kullanıldığı bir çalışmaya rastlanılmamıştır. Ayrıca gözden geçirmeye harcanan işgücü ve gözden geçirilen ürün tipi faktörlerinin hata yoğunluğuna etkisinin araştırılmadığı görülmüştür.

Bu çalışmada, gözden geçirilen ürün tipi, gözden geçirme için harcanan toplam işgücü ve ürün büyüklüğü faktörlerinin hata yoğunluğu üzerinde önemli bir etkisinin olup olmadığı araştırılmıştır.

Çalışmaya girdi teşkil eden veriler yazılım projeleri geliştiren bir Ar-Ge kurumunda yapılan gözden geçirme faaliyetlerinden toplanmıştır.

Verilerin toplandığı Ar-Ge kurumunda gözden geçirme sonuçlarının kaydedilmesi için üç farklı araç kullanılmaktaydı. Bu araçlar sırasıyla Atlassian JIRA, IBM JAZZ RTC ve Smartbear Collaborator araçlarıdır. Örnek olarak gözden geçirmelerde saptanan hataların kaydedildiği Smartbear Collaborator aracındaki kayıt formunun ekran görüntüsü Şekil 1'de gösterilmiştir.



Şekil 1. Hata kayıt formu.

Şekil 1'de gösterilen formda sırasıyla hatanın tanımı, önem derecesi (kritik, büyük, küçük) ve hata tipi (analiz, tasarım, dokümantasyon, kodlama, test, ara yüz, performans, sentaks vb.) bilgileri kaydedilmektedir.

2. Çok Etkenli Deney Tasarımı Yöntemi

Deney, çok etkenli tam rastgele düzene göre çözümlenmiştir [14]. Her gözede dört gözlem olan 3x2x2 çok etkenli bir deneydir. İki yönlü varyans analizi ve F testi kullanılarak 0,05 anlamlılık düzeyinde sonuçlar incelenmiştir.

2.1. Verilerin Toplanması

Yazılım projeleri geliştiren bir Ar-Ge kurumunda toplam yirmi (20) projeden örnekleme yöntemi ile veri toplanmıştır. Veri toplanan projelerin süreleri iki (2) ile beş (5) yıl arasında değişmekteydi.

Bu çalışmada plan, gereksinim ve tasarım tipindeki dokümanların gözden geçirme sonuçları kullanılmıştır. Toplanan veriler sırasıyla: gözden geçirmede bulunan hata sayısı, gözden geçirmeler için harcanan toplam işgücü, gözden geçirilen doküman sayfa sayısı ve gözden geçirilen doküman tipidir. Projelerde gözden geçirilen doküman büyüklükleri yirmi (20) ile üç yüz (300) sayfa arasında değişmekteydi. Gözden geçirilen dokümandaki toplam hata sayısının, gözden geçirilen dokümanın sayfa sayısına bölünmesi sonucunda elde edilen değer "hata yoğunluğu" dur. Bu ilişki Eşitlik (1)'de gösterilmiştir.

$$\text{Doküman hata yoğunluğu} = \frac{\text{Toplam hata sayısı}}{\text{Doküman sayfa sayısı}} \quad (1)$$

Gözden geçirme sonuçlarından elde edilen verilerin türleri deneyde kullanılacak faktörler seçilmiştir. Bu faktörlerle ilgili bilgiler sırasıyla belirtilmiştir:

a) Gözden geçirilen doküman büyüklüğü (sayfa sayısı). Gözden geçirilen dokümanlar için sayfa sayısı iki düzey olarak belirlenmiştir. Birinci düzey: kırk sayfadan az, ikinci düzey: kırk sayfadan fazla şeklinde belirlenmiştir.
b) Gözden geçirme için harcanan toplam işgücü (adamxdakika cinsinden gözden geçirme yapanların harcadıkları toplam işgücüdür). İş gücü için iki düzey belirlenmiştir. Birinci düzey: 300 adamxdakika dan az, ikinci düzey 300 adamxdakika dan fazla şeklinde belirlenmiştir.
c) Gözden geçirilen ürün tipi. Birinci düzey: planlar, ikinci düzey: gereksinim dokümanları, üçüncü düzey: tasarım dokümanları olarak belirlenmiştir.

Faktörlerin düzeyleri aşağıdaki yöntem ile belirlenmiştir:

Gözden geçirme sürecine tabi tutulan dokümanlar sayfa sayısına göre küçükten büyüğe doğru sıralanmıştır. Bu dokümanlara ait hata yoğunlukları ile Minitab aracında I-MR kontrol kartı oluşturulmuştur. Sonuç olarak 40 sayfadan az olan ve fazla olan dokümanlara ait hata yoğunluklarının kendi arasında homojen dağıldığı görülmüştür. Gözden geçirmeye harcanan işgücü büyüklükleri için de I-MR kontrol kartı oluşturulduğunda 300 adamxdakika değerinin altında ve üstündeki dokümanlara ait hata yoğunluklarının da kendi arasında homojen dağıldığı görülmüştür. Faktörlerin düzeyleri bu yöntem ile belirlenmiştir.

Tablo 1. Doküman tipi, işgücü ve büyüklük bilgilerine sınıflandırılmış hata yoğunlukları.

	0-300 adam*dk arası		**300 adam*dk dan fazla**	
	40 sayfadan az	**40 sayfadan fazla**	**40 sayfadan az**	**40 sayfadan fazla**
Plan dokümanları	2,90	0,20	5,46	2,44
	1,72	0,65	2,78	0,48
	1,91	0,23	1,62	1,66
	3,00	0,55	1,72	2,77
Gereksinim dokümanları	1,25	0,29	3,42	0,96
	0,93	2,11	2,72	1,36
	0,77	1,82	2,61	0,10
	2,28	1,97	1,89	2,40
Tasarım dokümanları	1,91	0,18	1,66	0,81
	0,77	1,32	1,23	0,11
	0,32	3,04	1,10	0,53
	0,34	2,05	0,83	0,65

Gözden geçirmelerden elde edilen hata yoğunluklarının doküman tipi, işgücü ve büyüklük faktörlerine göre sınıflandırılmış hali Tablo 1'de gösterilmiştir.

2.2. Veri Analizi

Yukarıdaki veriler ışığında aşağıdaki bölümlerde detayları verilen deney tasarımı çalışması gerçekleştirilmiştir:

2.2.1. Problem ve Değişkenlerin Tanımlanması

Doküman tipi, gözden geçirme için toplam harcanan işgücü ve doküman büyüklüğü faktörlerinin hata yoğunluğu üzerinde önemli bir etkisinin olup olmadığı araştırılmıştır.

2.2.2. Alınan Gözlem Sayısı ve Deneyin Sırası

Deneyde her göze için dörder veri toplanmıştır. Bu durumda toplam veri sayısı 3x2x2x4 = 48 olmuştur.

2.2.3. Kullanılacak Rasgeleleştirme Yöntemi

Her göze için alınan dörder veri tamamen rasgele olarak seçilmiştir.

2.2.4. Deneyi Tanımlayan Matematiksel Model

Bu deney düzeni için matematiksel model Eşitlik (2)'de verilmiştir;

$$Y_{ijkm} = \mu + A_i + B_j + AB_{ij} + C_k + AC_{ik} + BC_{jk} + ABC_{ijk} + \varepsilon_{m(ijk)} \quad (2)$$

Burada;

Y_{ijkm}: j. İşgücü, k. Doküman büyüklüğü ile i. Doküman tipindeki özelliklerine sahip hata yoğunluğudur.

μ : Tüm gözlemlerde ortak etkiyi (örneklenen kitlenin gerçek ortalama hata yoğunluğunu) göstermektedir.

A_i: Doküman Tipi i = 1, 2, 3

B_j: Gözden geçirme toplam işgücü j = 1, 2

C_k: Gözden geçirilen doküman büyüklüğü k = 1, 2

AB_{ij} : A ve B arasındaki etkileşim

AC_{ik} : A ve C arasındaki etkileşim

BC_{jk} : B ve C arasındaki etkileşim

ABC_{ijk} : A, B ve C arasındaki etkileşim

$\varepsilon_{m(ijk)}$: (i, j, k) gözesindeki rasgele hata

2.2.5. İki Yönlü Varyans Analizi

Gözden geçirmelerden elde edilen hata yoğunluk bilgileri üzerinde çok etkenli tam rastgele düzene göre çözümleme yapılmıştır. Bu düzen tam rasgele olarak hazırlanmış, her gözede 4 gözlem bulunan 3x2x2 çok etkenli bir deneydir. Bu deney için veriler Tablo 1'de verilmiştir. Tablo 1'deki veriler kodlanarak Tablo 2 elde edilmiştir.

Tablo 2. Tablo 1'deki verilerde (X-48) kodlama sonuçları.

	0-300 adam*dk arası				**300 adam*dk dan fazla**				
	40 sayfadan az		**40 sayfadan fazla**		**40 sayfadan az**		**40 sayfadan fazla**		
Plan dokümanları	2,90		0,20		5,46		2,44		
	1,72	24,01	0,65	0,82	2,78	43,12	0,48	16,61	30,09
	1,91		0,23		1,62		1,66		
	3,00		0,55		1,72		2,77		
		9,53		1,63		11,58		7,35	
Gereksinim dokümanları	1,25		0,29		3,42		0,96		
	0,93	8,21	2,11	11,72	2,72	29,47	1,36	8,54	26,88
	0,77		1,82		2,61		0,10		
	2,28		1,97		1,89		2,40		
		5,23		6,19		10,64		4,82	
Tasarım dokümanları	1,91		0,18		1,66		0,81		
	0,77	4,46	1,32	15,21	1,23	6,17	0,11	1,37	16,85
	0,32		3,04		1,10		0,53		
	0,34		2,05		0,83		0,65		
		3,34		6,59		4,82		2,10	
Genel	**18,10**		**14,41**		**27,04**		**14,27**		**T..=73,82**

Çok etkenli tam rastgele düzene göre çözümleme:

a = 3, b = 2, c=2, n=4 olmak üzere ilgili hesaplamalar aşağıda Eşitlik (3-11)'de sırasıyla gösterilmiştir.

$$KT_{genel}=\sum_{i}^{a}\sum_{j}^{b}\sum_{k}^{c}\sum^{n}Y_{ijk}^2-\frac{T_{..}^{\ 2}}{nabc}$$

$$=169,71-\frac{(73,82)^2}{48}=56,18 \quad (3)$$

Yukarıdaki denklemde;

Y_{ijk}^2 :Tüm gözlem değerlerinin kareler toplamı

$T_{..}$: Faktörlerin sütun ve satır toplamı

n: Gözlem tekrar sayısı

a: A_i faktörü düzey sayısı

b: B_j faktörü düzey sayısı

c: C_k faktörü düzey sayısı

olmak üzere nabc = 4*3*2*2 = 48 değeri deneydeki toplam gözlem sayısını vermektedir.

$$KT_A=\sum_{i}^{a}\frac{T_{i.}^2}{nb}-\frac{T_{..}^2}{nab}$$

$$=\frac{(30,09)^2+(26,88)^2+(16,85)^2}{16}-\frac{(73,82)^2}{48}=5,96 \quad (4)$$

$$KT_B=\frac{(32,51)^2+(41,31)^2}{24}-\frac{(73,82)^2}{48}=1,61 \quad (5)$$

$$KT_C=\frac{(45,14)^2+(28,68)^2}{24}-\frac{(73,82)^2}{48}=5,64 \quad (6)$$

$$KT_{BXC}=\frac{(18,1)^2+(14,41)^2+(27,04)^2+(14,27)^2}{12}-$$

$$\frac{(73,82)^2}{48}-1,61-5,64=1,73 \quad (7)$$

$$KT_{BXA}=\frac{(11,16)^2+(11,42)^2+(9,93)^2+(18,93)^2+(15,46)^2}{8}+$$

$$\frac{(6,92)^2}{8}-\frac{(73,82)^2}{48}-1,61-5,96=3,75 \quad (8)$$

$$KT_{CXA}=\frac{(21,11)^2+(8,98)^2+(15,87)^2+(11,01)^2+(8,16)^2}{8}+$$

$$\frac{(8,69)^2}{8}-\frac{(73,82)^2}{48}-5,64-5,96=5,05 \quad (9)$$

$$KT_{AXBXC}=\frac{(9,53)^2+(1,63)^2+\ldots+(4,82)^2+(2,1)^2}{4}-\frac{(73,82)^2}{48}$$

$$-1,61-5,64-5,96-1,73-3,75-5,05=4,21 \quad (10)$$

$$KT_{hata}=KT_{genel}-KT_B-KT_C-KT_A-KT_{BXC}-KT_{BXA}-KT_{CXA}-$$

$$-KT_{AXBXC}=56,18-1,61-5,64-5,96-1,73-3,75$$

$$-4,21=28,23 \quad (11)$$

Elde edilen değerler varyans analizi tablosunda yerine konularak kareler toplamının serbestlik derecelerine

bölünmesi ile elde edilen kareler ortalamaları Tablo 3'te gösterilmiştir.

3. Bulgular ve Tartışma

Bu bölümde hipotez testleri ile hata yoğunluğuna etki eden faktörler araştırılmıştır. Bulunan sonuçlar Tablo 3'te gösterilmiştir.

Tablo 3. Serbestlik dereceleri ve kareler ortalamaları.

Değişkenlik kaynağı	sd	KT	KO
B işgücü	1	1,61	1,61
C büyüklük	1	5,64	5,64
A tip	2	5,96	2,98
BXC	1	1,73	1,73
BXA	2	3,75	1,88
CXA	2	5,05	2,53
AXBXC	2	4,21	2,10
Hata	36	28,23	0,78
Genel	47		

$$F_B=\frac{1,61}{0,78}=2,06<4,12=F_{1,36,0,05} \quad (12)$$

H_0 Kabul, gözden geçirme için harcanan iş gücünün hata yoğunluğuna etkisi yoktur.(Eşitlik (12)).

$$F_C=\frac{5,64}{0,78}=7,23>4,12=F_{1,36,0,05} \quad (13)$$

H_0 Ret, gözden geçirilen doküman büyüklüğünün hata yoğunluğuna etkisi ***vardır*** (Eşitlik (13)).

$$F_A=\frac{2,98}{0,78}=3,82>3,27=F_{2,36,0,05} \quad (14)$$

H_0 Ret, gözden geçirilen doküman tipinin hata yoğunluğuna etkisi ***vardır*** (Eşitlik (14)).

$$F_{BXC}=\frac{1,73}{0,78}=2,21<4,12=F_{1,36,0,05} \quad (15)$$

H_0 Kabul, "işgücü-büyüklük" etkileşiminin hata yoğunluğuna etkisi yoktur (Eşitlik (15)).

$$F_{BXA}=\frac{1,88}{0,78}=2,41<3,27=F_{2,36,0,05} \quad (16)$$

H_0 Kabul, "işgücü-tip" etkileşiminin hata yoğunluğuna etkisi yoktur (Eşitlik (16)).

$$F_{CXA}=\frac{2,53}{0,78}=3,24<3,27=F_{2,36,0,05} \quad (17)$$

H_0 Ret, "büyüklük-tip" etkileşiminin hata yoğunluğuna etkisi yoktur (Eşitlik (17)).

$$F_{AXBXC}=\frac{2,10}{0,78}=2,69<3,27=F_{2,36,0,05} \quad (18)$$

H_0 Kabul, "işgücü-tip-büyüklük" etkileşiminin hata yoğunluğuna etkisi yoktur (Eşitlik (18)).

Bu çalışmada kullanılan çok etkenli tam rasgele deney düzeni yöntemi, yazılım kod hata yoğunluğuna etki eden faktörlerin araştırılmasında da kullanılabilir. Kod hata yoğunluğuna etki eden faktörler olarak gözden geçirilen kod büyüklüğü, gözden geçirme için harcanan işgücü, gözden geçirmeye katılanların tecrübesi düzeyi deneye girecek faktörler olarak kabul edilip kod hata yoğunluğuna etkisinin olup olmadığı araştırılabilir.

4. Sonuçlar

Deneyde göz önüne alınmayan ancak deney sonuçlarını etkileyebilecek olan birçok değişkenin etkisini ortalama düzeye getirmek için rastgeleleştirme yöntemi kullanılmıştır.

Yapılan çalışmada gözden geçirmeler için üç faktör, iki düzeyli ve dörder tekrarlı deney tasarımı gerçekleştirilmiştir. Hipotez testleri ile sonuçlar yorumlandığında; doküman gözden geçirmelerinde doküman büyüklüğü ve doküman tipi faktörlerinin hata yoğunluğuna anlamlı etkisinin olduğu, doküman gözden geçirme için harcanan işgücü faktörünün hata yoğunluğuna anlamlı etkisinin olmadığı görülmüştür.

Bu çalışma, yazılım ürünün kalite göstergelerinden biri olan gözden geçirme hata yoğunluğuna anlamlı derecede etki eden faktörlerin hangileri olduğunu ortaya çıkarmıştır. Bu çalışma temel alınarak sonraki çalışmalarda, yazılım test sürecinden elde edilen test hata yoğunluğuna etki eden faktörler belirlenebilir. Test hata yoğunluğuna etki eden faktörler olarak test edilen yazılımın kod büyüklüğü, test için harcanan işgücü, test yapanların tecrübesi deneye girecek faktörler olarak kabul edilebilir.

Kaynaklar

[1] IEEE Standard 1028-2008 for Standard for Software Reviews and Audits.

[2] IEEE Standard 1012-2012 for Standard for System and Software Verification and Validation.

[3] CMMI Model (Capability Maturity Model Integration) Development v2.0.

[4] Wedyan F., Salameh H. B., Al-Ajloni W., Al-Manai S., 2015. Investigating the Relationship between Software Defect Density and Cost Estimation Drivers: An Empirical Study. Journal of Theoretical and Applied Information Technology, **82**(3), 446-453.

[5] Zhang W., Ma Z., Zhang Wenge, Lu Q., Nie X., 2015. Correlation Analysis of Software Defects Density and Metrics. Applied Mechanics and Materials, **713-715**, 2225-2228.

[6] Bosu A., Greiler M., Bird C., 2015. Characteristics of Useful Code Reviews: An Empirical Study at Microsoft. 12th Working Conference on Mining Software Repositories, Florence/Firenze, 16-24 May, 146-156.

[7] Mandhan N., Verma D. K., Kumar S., 2015. Defect Density using Static Metrics. International Conference on Computing, Communication and Automation, Greater Noida-India, 15-16 May, DOI: 10.1109/CCAA.2015.7148499.

[8] Al-Jamimi H. A., 2016. Toward Comprehensible Software Defect Prediction Models Using Fuzzy Logic. 7th IEEE International Conference on Software Engineering and Service Science (ICSESS), Beijing-China, 26-28 August, 127 – 130.

[9] Shah S. M. A., Morisio M., Torchiano M., 2013. Software Defect Density Variants: A Proposal. 4th International Workshop on Emerging Trends in Software Metrics (WETSoM), San Francisco-USA, 21 May, 56 – 61.

[10] Chen J., Yang Y., Hu K., Xuan Q., Liu Y., Yang C., 2019. Multiview Transfer Learning for Software Defect Prediction. IEEE Journals & Magazines, **7**, 8901 – 8916.

[11] Xiao P., Yan X., Huang F., 2017. How Domain Knowledge Accumulation Influences Software Defects: An Empirical Analysis. IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C), Prague, 25-19 July, 24 – 30.

[12] Yamashita K., Huang C., Nagappan M., Kamei Y., 2016. Thresholds for Size and Complexity Metrics: A Case Study from the Perspective of Defect Density. IEEE International Conference on Software Quality, Reliability and Security (QRS), Vienna, 1-3 August, 191 – 201.

[13] Alfadel M., Kobilica A., Hassine J., 2017. Evaluation of Halstead and Cyclomatic Complexity Metrics in Measuring Defect Density. 9th IEEE-GCC Conference and Exhibition (GCCCE), Manama-Bahrain, 8-11 May, 1 – 9.

[14] Hicks, C. R., 1973. Deney Düzenlemede İstatistiksel Yöntemler, Holt, Rinehart and Winston, United States of America.