Mobil Uygulamaların Kalite Özelliklerinin Ölçümü

Nurseda ÖZDEMİR, Bahar GEZİCİ, Kıvanç DİNÇER

Hacettepe Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Anabilim Dalı
Beytepe Kampüsü, 06800 Ankara

nursedaozdemr@gmail.com, bahargezici@hacettepe.edu.tr,
kivanc.dincer@hacettepe.edu.tr

Özet. Yazılımların (dışsal) kalite özelliklerinin doğrudan ölçülmesi kolay olmadığından, bunların değerlendirilmesi içsel kalite özelliklerini ölçen yazılım metrikleri vasıtasıyla yapılmaktadır. Bu çalışmada amacımız, üzerinde henüz çok fazla çalışma bulunmayan bir konu olan, bir yazılımın masaüstü platformlar için geliştirilmiş sürümü ile mobil sürümü arasında kalite özelliklerinin ne derece benzediğinin araştırılmasıdır. Bu soruyu cevaplandırabilmek için AdBlock, KeePass, Telegram ve Zulip adlı açık kaynak uygulamaların masaüstü ve mobil versiyonlarının içsel kalite özellikleri karşılaştırılmıştır. Önce "Understand" statik kod analiz aracı kullanılarak uygulamaların masaüstü ve mobil versiyonları için Chidamber & Kemerer (CK) metrik kümesinden DIT, PLOC, WMC, CBO, NOC, ve RFC metriklerinin değerleri ölçülmüştür. Sonra bu iki platform için elde edilen metrik değerleri karşılaştırmalı olarak analiz edilmiştir.

Anahtar Kelimeler: Mobil Uygulamaların Kalite Özellikleri, Kalite Özelliklerinin Ölçümü, Nesne Yönelimli C&K Metrikleri.

Abstract. Since it is difficult to measure the (external) quality attributes of a software product directly, an evaluation is made by using some software metrics that represent the value of related internal quality attributes. The objective of this work is to answer the research question – which has not been studied much in the literature yet – of whether the quality attributes of an application will be similar in its desktop and mobile versions. Four open-source applications having both desktop and mobile versions have been studied: AdBlock, KeePass, Telegram and Zulip. First, the source code of these applications was evaluated using Chidamber & Kemerer (CK) object-oriented metric set/suite. DIT, PLOC, WMC, CBO, NOC and RFC metrics from the CK metric set were collected for both versions using the "Understand" static code analysis tool. Then the measurement results for both applications have been reviewed by a comparative analysis to find an answer for our research question.

Keywords: Mobile Software Quality Attributes, Measuring Quality Attributes, Object-Oriented C&K Metric Suite.

1 Giriş

Teknolojinin hızla geliştiği, hayatı kolaylaştırıcı cihazların sayısının her geçen gün arttığı günümüzde, mobil cihazlar her alanda karşımıza çıkmaktadır. Bu gelişme mobil uygulamaların sayısındaki artışı da beraberinde getirmiştir. Haziran 2015 itibariyle Google Play'de 1,6 milyon, Apple App Store'da 1,5 milyon, Amazon App Store'da 400 bin ve Windows Phone Store'da 340 bin mobil uygulama bulunmaktadır [1].

Mobil uygulamaların son derece popüler olması, mobil uygulama geliştirmenin popülerliğini de tetiklemiştir. Ancak mobil uygulamaların satış fiyatlarının masaüstü uygulamalara göre daha düşük olması mobil uygulama geliştiricilerinin geliştirme ve bakım maliyetlerini daha düşük tutmalarını gerektirmektedir. Dolayısıyla mobil uygulama geliştiricisi tekrar kullanılabilirliği yüksek, gelecekte meydana gelebilecek değişikliklere uyum sağlayabilen, geliştirmesi, değiştirmesi ve bakımı kolay uygulamalar geliştirmeyi hedeflemelidir. Geliştiricinin bu hedefe ulaşması yazılımının "kaliteli" olması ile mümkün olmaktadır. Yazılımın kalitesi genel olarak gereksinimlere veya beklentilere uygunluk olarak tanımlanmıştır, bu bağlamda kaliteye geliştirici perspektifinden baktığımızda kalite beklentileri geliştirme ve bakım maliyetlerinin düşük olması olarak karşımıza çıkabilir.

Kalite kavramı her ne kadar öznel bir kavram olsa da çeşitli kalite modelleriyle ortak bir pencere yakalanabilmektedir. Örneğin ISO 9126 yazılım kalite değerlendirmesi için kullanılan uluslararası bir standarttır ve kaliteyi işlevsellik, güvenilirlik, kullanılabilirlik, verimlilik, bakım yapılabilirlik, taşınabilirlik gibi sınıflara ayırmıştır [2]. Fenton ve arkadaşları [3] yazılım sonucu ortaya çıkan ürünün kalite özelliklerini içsel kalite özellikleri ve dışsal kalite özellikleri olarak ikiye ayırmıştır. *İçsel kalite özellikleri*; çeşitli metrikler ile yazılımın direk kendisinden (kaynak kodundan) ölçülebilen özellikleridir. Örneğin iki sınıf arasındaki bağımlılığı ölçmek istediğimizde direk olarak programın kaynak kodunu analiz ederek bu bilgiye ulaşabiliriz. *Dışsal kalite özellikleri* ise; ürünle ilgili geliştiricilerin veya kullanıcıların kanaatlerini yansıtır. Burada ürünün nasıl davrandığı ön plana çıkan etkendir. Örneğin kullanıcı dostu olma, bakım yapılabilirlik, güvenilirlik gibi.

Yazılımların dışsal kalite özelliklerinin doğrudan ölçülmesi kolay olmadığından, bunların değerlendirilmesi içsel kalite özelliklerini ölçen yazılım metrikleri vasıtasıyla yapılmaktadır [4]. Bu yöntem sıklıkla alternatif yazılımların değerlendirilmesi ve seçimi için kullanılmaktadır [5].

Bazı araştırmacılar kalite ölçümleri için çeşitli metrik setleri yayınlamışlardır. Bu metrik setlerinden bazıları Chidamber & Kemerer (CK) Metrik Kümesi [6], Brito e Abreu MOOD Metrik Kümesi [7], Bansiya ve Davis QMOOD Metrik Kümesi [8] vb.dir [9]. Bu metrik setleri geleneksel yazılımlara defalarca uygulanmış ve onaylanmıştır. Ancak hızla yayılmakta olan mobil yazılımların (dışsal) kalite özelliklerini belirleyen içsel (kalite) özelliklerinin masaüstü sürümlerine göre benzerlik taşıyıp taşımadığı konusu üzerinde literatürde çok fazla çalışma bulunmamaktadır. Bu çalışmada amacımız (içsel) kalite özelliklerinin aynı yazılımın mobil ve masaüstü platformlar için geliştirilen sürümlerinde nasıl değişiklik göstereceğinin tespit edilmesidir.

Araştırma Sorusu: Bir yazılımın masaüstü platformlar için geliştirilmiş sürümünün kalitesi mobil sürümünün kalitesini belirleyici midir? Veya bir yazılımın masaüstü ve mobil sürümü kalite özellikleri açısından ne derece benzerlik taşır?

Bu araştırma sorusuna cevap bulabilmek için hem masaüstü uygulaması hem mobil uygulaması bulunan dört açık-kaynak uygulama incelenmiştir. Bu uygulamalar AdBlock[10], KeePass[11], Telegram[12] ve Zulip[13] isimli uygulamalardır. Araştırmamız CK metrik kümesinden seçilen 6 temel metriğin bu uygulamaların masaüstü ve mobil versiyonları için toplanmasını ve bu iki platform için elde edilen ölçüm sonuçlarının karşılaştırmalı olarak analizini içermektedir.

Bu bildirinin devamı şu şekilde yapılandırılmıştır. İkinci bölümde ilgili çalışmalar sunulmuştur. Üçüncü bölüm CK metrik kümesine ait metriklerin açıklamalarını içermektedir. Dördüncü bölüm araştırma yönteminin detaylarını ve tasarımı içermektedir. Beşinci bölümde yapılan çalışmadan çıkan sonuçların analiz ve tartışması, altıncı bölümde ise sonuç ve gelecek çalışmalar sunulmaktadır.

2 İlgili Çalışmalar

Literatürde yazılım ölçümü ve metrikler birçok çalışmanın konusu olmuştur. Ancak mobil yazılımlarda geleneksel metriklerin uygulanması ile ilgili çok fazla çalışma bulunmamaktadır. Tablo 1'de araştırma yapılan veri tabanları ve anahtar kelimelerle eşleşen makale sayıları verilmiştir. İlgili anahtar kelimeler makalenin başlığında, özetinde ve anahtar kelimelerinde aranmıştır. Bu makalelerden hem mobil uygulamalar üzerinde çalışılmış olması hem de çalışmalarda yer alan yazılım metriklerinin bizim uygulamamızla benzerlik taşıması açısından çalışmamızla ilgili olan iki tanesine özet olarak değinilmiştir.

Tablo 1. Online veritabanları ve bulunan makale sayıları

Online Veritabanı	"object oriented metrics" & "mobile applications"	"software metrics" & "mobile applications"	"software metrics" & "mobile"
Scopus	2	9	32
SpringerLink	7	41	324
Web of Science	0	2	11
Elsevier	1	24	151
Emerald	0	1	1
Wiley InterSciene	1	8	80
ACM	0	1	3
IEEEXplore	1	8	62

Jošt, Huber & Hericko [14] 2013 yılında yaptıkları çalışmada geleneksel yazılım metriklerinin mobil uygulamalarda uygulanabilirliğini araştırmışlardır. Bu kapsamda 3 farklı mobil uygulama platformu (Android, iOS ve Windows Phone) için çok fazla iş kuralı içermeyen, grafiksel arayüz eleman sayısı az olan, genel olarak sınırlı sayıda dosya içeren küçük ölçekli bir yazılım geliştirmişler ve her uygulama için *Kalıtım ağacının derinliği (DIT)*, *Alt sınıf sayısı (NOC)*, *Metotlardaki uyum eksikliği* (LCOM), *Sınıfın ağırlıklı metot sayısı (WMC)*, *Metot sayısı (NOM)*, *Sınıf sayısı*

(NOC2) ve *Çevrimsel karmaşıklık (CC)* metriklerini üç farklı platform için hesaplamışlardır. Yapılan analizler sonucunda ilgili metriklerin mobil uygulamalar için kullanılabilirliği kabul edilebilir seviyededir sonucuna varılmıştır. Ayrıca işlevsel olarak eşdeğer uygulamalardan toplanan metriklerin farklı platformlar için farklılaşacağı da bu çalışmanın sonuçları arasındadır [14].

2015 yılında Syer, Nagappan ve arkadaşları yaptıkları çalışmada [15] farklı platformlarda geliştirilen mobil yazılımların kod kalitesini araştırmışlardır. Bu amaçla beş farklı mobil uygulama incelenmiştir. Bunlar ConnectBot, FBReader, KeePassDroid, Sipdroid, XBMCRemote Android mobil uygulamalarıdır. Bu çalışmada analiz edilen metrikler ise Kod Satır Sayısı (Lines of Code – LOC), Coupling, Cohesion ve Platform metrikleridir. Coupling metriği CK metriklerinden CBO metriği ile, Cohesion metriği ise LCOM metriği ile eşleşmektedir. Platform metriği bu çalışmaya özgü hesaplanan platform bağımlılık oranını ifade eden ölçümdür. Sunulan sonuç platform metriğinin kaynak kod kalitesi üzerinde en tutarlı etkiye sahip olduğunu göstermiştir.

İncelenen ilgili çalışmalara bakıldığında üzerinde çalışılan metriklerin genel olarak CK metrik kümesinden (LOC, WMC, Coupling, Cohesion, …) seçildiği gözlenmiştir. Ayrıca farklı platformda geliştirilen mobil uygulamalar için ilgili metrikler kendi içlerinde karşılaştırılmış olup mobil uygulamalarda her bir metriğin göstermiş olduğu davranış karşılaştırılmıştır. Fakat incelenen hiçbir çalışmada uygulamaların mobil ve masaüstü versiyonları için herhangi bir kıyaslamaya gidilmemiştir.

3 Chidamber & Kemerer (CK) Nesne Tabanlı Metrik Kümesi

Nesne yönelimli yazılımlar kapsülleme (*encapsulation*), kalıtım (*inheritance*), çok biçimlilik (*polymorphism*), uyumluluk (*coherence*) ve bağımlılık (*coupling*) gibi farklı özelliklere sahiptir [16]. Bu özellikler yazılım kalitesinde de etki göstermektedir. Gerek geliştirme, gerekse test ve bakım faaliyetlerinde bu özellikler kullanılarak çıkarım yapılabilmektedir. Kullanımı hızla artan bu dillerdeki yazılımların kalite ölçüm metriklerinin diğer yazılımlardan farklı olması sebebiyle Chidamber ve Kemerer [6] 1994 yılında kendi adlarını taşıyan (CK) bir metrik kümesi oluşturmuşlardır. CK metrik kümesi temelde aşağıda tanımlanan altı metriği ele almaktadır.

3.1 WMC: Sınıf Başı Ağırlıklı Metot Sayısı (*Weighted Methods Per Class*)

Sınıf başına düşen ağırlıklı metotlardır. Eğer bir C sınıfının metotları sırasıyla M1, M2, … Mn ise ve bu metotların karmaşıklığı c_1, …, c_n ise bu C sınıfının ağırlıklı metot sayısı (WMC) aşağıdaki gibidir [17]. Sınıfların olabildiği kadar öz olarak tasarımı beklendiğinden sınıfın karmaşıklık derecesini ifade eden bu metriğin değerinin düşük olması beklenmektedir.

$$\text{WMC} = \sum_{i=1}^{n} ci$$

3.2 DIT: Kalıtım Ağacının Derinliği (*Depth Of Inheritance Tree*)

Bir sınıfın kalıtım ağacının köküne olan maksimum uzaklığıdır. Eğer sınıf herhangi bir sınıftan türetilmemişse değeri 0'dır [6, 16, 18]. Tekrar kullanılabilirlik açısından bakıldığında bu metrik değerinin yüksek olmasının getirdiği kazanımlar olmasına rağmen, karmaşıklık açısından değerlendirildiğinde ise düşük değerli olması beklenmektedir. Tekrar kullanılabilirlik ve karmaşıklık için ortalama (dengede) bir değerde bulunması beklenmektedir.

3.3 NOC: Alt Sınıf Sayısı (*Number Of Children*)

Bir sınıftan doğrudan türetilmiş olan sınıf sayısıdır. Alt sınıf sayısının yüksek olması genel olarak daha karmaşık, bakım yapılabilmesi daha zor ve hataya eğilimi daha yüksek kod anlamına gelmektedir. Bu sebeple değeri düşük olmalıdır [6, 17, 18].

3.4 CBO: Nesne Sınıfları Arasındaki Bağımlılık (*Coupling Between Object Classes*)

Bir sınıfın bağımlı olduğu sınıf sayısıdır. Bir sınıf içindeki değişken veya metodun diğer sınıf tarafından kullanılması iki sınıf arasında bağımlılık oluşturmaktadır [6, 16, 18]. Kapsülleme ve modülerliğin yüksek olması açısından bakıldığında bu metriğin düşük değerde olması beklenmektedir.

3.5 RFC: Sınıfın Tetiklediği Metot Sayısı (*Response For a Class*)

Bir sınıftaki metotların sayısı ve bu sınıfın metotları tarafından çağrılan sınıfların metot sayısının toplamıdır. Yani bir nesne tarafından çağrılabilen metot sayısıdır. Test edilebilirlik ve bakım açısından bu metrik değerinin yüksek olması karmaşıklığı artırmaktadır. Bu sebeple değerinin düşük olması beklenmektedir [6, 16, 18].

3.6 LCOM: Metot İçi Uyumsuzluk (*Lack of Cohesion in Methods*)

Bir C sınıfının n tane metodu varsa M_1, M_2, … M_n, $\{I_j\}$ seti M_i tarafından kullanılan nitelik değişkenleri *(instance variables)* kümesiyse; LCOM bu n kümenin kesişimi sonucu oluşan ayrık küme sayısıdır [6, 16, 18]. Yüksek kaliteli bir yazılımda LCOM değerinin düşük olması beklenmektedir. Bu değerin yüksek olması karmaşıklığı artıracaktır ve böyle bir sınıfın iki veya daha fazla alt sınıfa bölünmesi önerilmektedir.

4 Araştırma Yöntemi

Bu bölümde nesne tabanlı yazılım metriklerinden olan CK metrik kümesinin mobil uygulamalar için kullanılabilirliğinin araştırılması için takip edilen yaklaşım özetlenmiştir. Öncelikle analiz için veri sağlayacak olan masaüstü ve mobil uygulamalar seçilmiştir. İkinci olarak araştırmada kullanılacak metrik seti seçilmiştir. Daha sonra

seçilen masaüstü ve mobil uygulamalarda statik kod analizleri yapılarak ilgili metrikler çıkarılmıştır. Son olarak hesaplanan veriler analiz edilerek mobil uygulamalarda nesne tabanlı yazılım metriklerinin kullanılıp kullanılamayacağı yorumlanmıştır.

4.1 Uygulama seçimi

Bu çalışmada statik kod analizi yapılan uygulamalar aşağıdaki kriterler göz önüne alınarak seçilmiştir:

- *Açık kaynak kodlu uygulamalar.* Seçilecek uygulamaların kaynak kodlarına Github tarzı veri depolarından erişim sağlayabilmek için uygulamaların açık kaynak kodlu olması gerekmektedir.
- *Mobil ve Masaüstü platformlar.* Seçilecek uygulamalarda karşılaştırma yapabilmemiz için uygulamaların her iki farklı platform için geliştirilmiş olması gerekmektedir. Çoğu mobil uygulama, masaüstü uygulaması bulunmamasından dolayı bu çalışmada kullanılamamıştır.
- *Nesne tabanlı dillerde geliştirilmiş uygulamalar.* Seçilecek uygulamalarda CK nesne tabanlı metrik kümesi kullanılacağı için uygulamaların nesne yönelimli bir programlama dilinde yazılması gerekmektedir. Örneğin masaüstü uygulaması C dilinde yazılmış mobil uygulamalar bu kıstasa uymadığından elenmiştir.
- *Kod bazının basit olması.* Seçilecek uygulamanın mobil ve masaüstü veri depolarının ayrı olması gerekmektedir. Masaüstü ve mobil kaynak kodlarının iç içe yayınlandığı kod depoları ayrıştırmayı zorlaştırdığı için filtrelenmiştir.

Ek olarak, yukarıdaki kriterlere uyan ancak kod büyüklüğü çok olmayan (örneğin uygulama geliştirme aşamasındayken yapılan değişiklik sayısı az olan, koda katkıda bulunan kişi sayısı az olan), az sayıda sınıf içeren uygulamalar da filtrelenmiştir.

Tablo 2 ilgili kriterlerden geçen ve bu çalışmada kullandığımız son uygulama listesini içermektedir.

Tablo 2. Analiz edilen uygulamalar (Son kaynak kod erişim tarihi: Mayıs 2016)

Uygulama	**Prg.Dili**	**Kaynak Kod Adresi**
KeePass	C++	https://github.com/keepassx/keepassx
KeePassDroid	Java	https://github.com/bpellin/keepassdroid
AdBlockPlus	Java	https://github.com/adblockplus/adblockplussbrowser
AdBlockPlus Mobil	Java	https://github.com/adblockplus/adblockplusandroid
Telegram Masaüstü	C++	https://github.com/telegramdesktop/tdesktop
Telegram Mobil	Java	https://github.com/DrKLO/Telegram
Zulip Masaüstü	C++	https://github.com/zulip/zulip-desktop
Zulip Mobil	Java	https://github.com/zulip/zulip-android

4.2 Statik kaynak kod analizi aracı seçimi

Seçilen her masaüstü ve mobil uygulamada statik kaynak kod analizi Understand [19] analiz aracıyla yapılmıştır. Understand, Java, C++, C#, Ada, Python gibi çeşitli programlama dillerinde yazılmış kaynak kodunu ölçmek ve analiz etmek için kullanılan, Scitools tarafından geliştirilen bir ticari statik analiz araç setidir. Aralarında NASA, US Navy, UK Ministry of Defence, Apple, Microsoft, ve Google gibi firmaların da bulunduğu yaklaşık 20.000 kullanıcısı vardır [19]. Güncel olarak geliştirilmeye devam etmektedir ve son sürümü 3 Haziran 2016 tarihinde yayınlanmıştır [20].

Araştırma için seçilen uygulamalar Understand statik kod analiz aracı kullanılarak analiz edilmiş ve bu uygulamalarda CK metrikleri olarak adlandırılan altı metrik (WMC, DIT, NOC, CBO, RFC, LCOM) hesaplanmıştır [6].

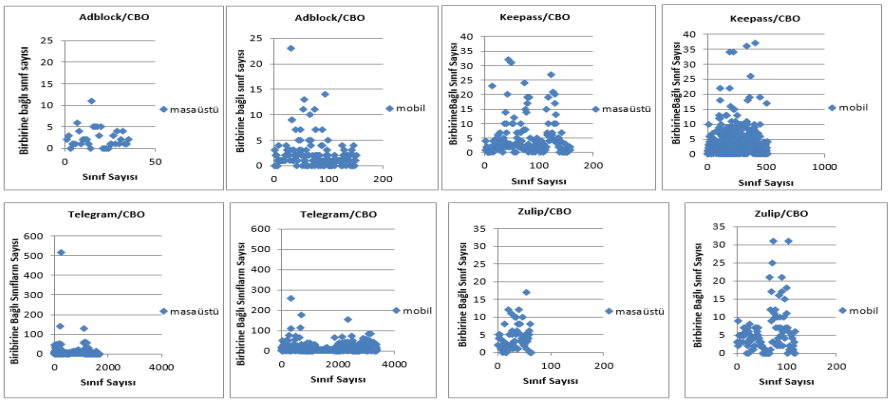
5 Sonuçların Analizi ve Tartışma

Seçilen dört farklı uygulamanın masaüstü ve mobil versiyonları üzerinde yapılan inceleme ve analizlerin sonuçları altı metrik ile ilişkili çeşitli grafikler ve tablolar olarak özetlenmiştir. Bu çıktıların değerlendirilmesi aşamasında, yapılan çalışmanın amacına uygun olarak geleneksel metriklerin mobil yazılımlarda da kullanılabilirliği gözlemlenmeye çalışılmıştır. Metriklerin sınıf seviyesinde değerlendirmeye tabi tutulması dolayısıyla her bir uygulamanın sınıf sayısı Tablo 3'te gözlenmektedir:

Tablo 3. Analiz edilen uygulamaların sınıf sayıları

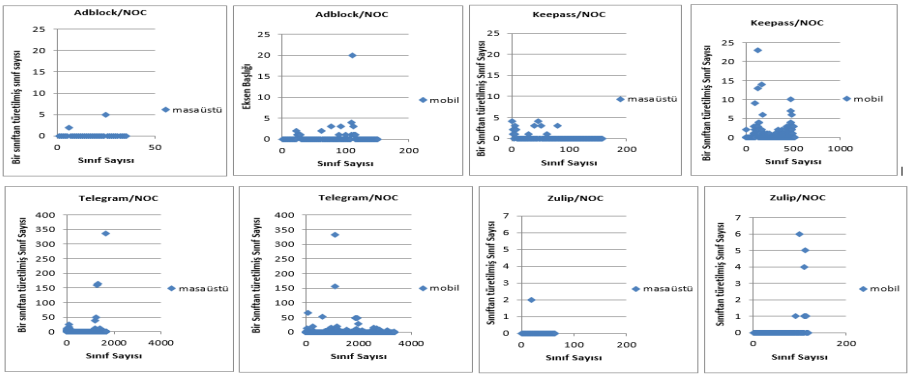
Uygulama Adı	Versiyon	Sınıf Sayısı
AdBlock	masaüstü	36
	mobil	152
KeePass	masaüstü	160
	mobil	510
Telegram	masaüstü	1661
	mobil	3354
Zulip	masaüstü	65
	mobil	119

Nesne Sınıfları Arasındaki Bağımlılık (CBO) metriği ile bir sınıfın bağımlı olduğu sınıf sayısının değerinin ölçümü yapılmış olup, elde edilen grafikler ile tüm uygulamaların mobil versiyonlarının x ekseni doğrultusunda bir genişlemeye neden olan değerlere sahip olduğu görülmektedir. Yazılım kalitesi açısından CBO değerinin düşük olması beklenmektedir. Dolayısıyla elde edilen bulgular ile AdBlock ve KeePass uygulamalarının mobil versiyonlarında daha düşük değerler karşımıza çıkarken; Telegram ve Zulip uygulamalarının masaüstü versiyonlarında ölçülen bu metrik değerinin daha düşük olduğu görülmüştür (Şekil 1).



Şekil. 1. CBO Metrik Değeri vs. Sınıf Sayısı

Alt Sınıf Sayısı (NOC) metriği incelendiğinde dört uygulamanın üçünün (AdBlock, KeePass, Zulip) masaüstü versiyonlarında, Telegram'ın ise mobil versiyonunda bu değerin daha düşük olduğu gözlenmiştir. Ayrıca, genelde mobil versiyonlarda bir sınıftan doğrudan türetilmiş sınıf sayısının fazla olduğu göze çarpmıştır (Şekil 2).

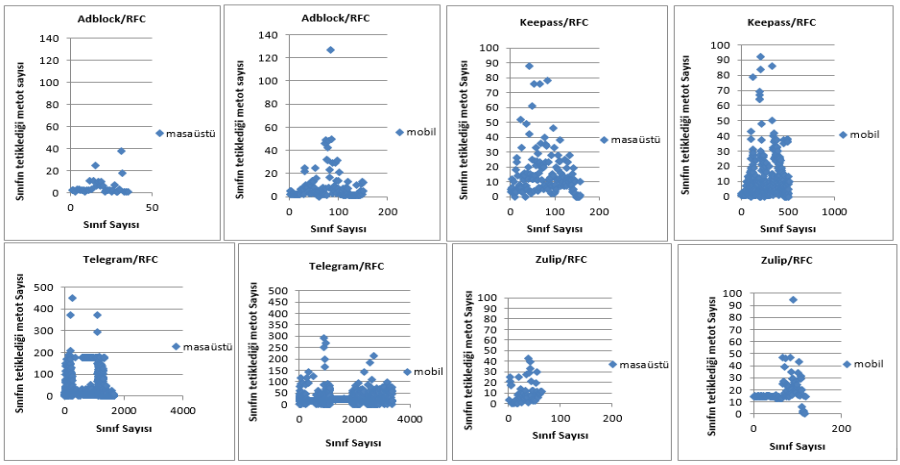


Şekil. 2. NOC Metrik Değeri vs. Sınıf Sayısı

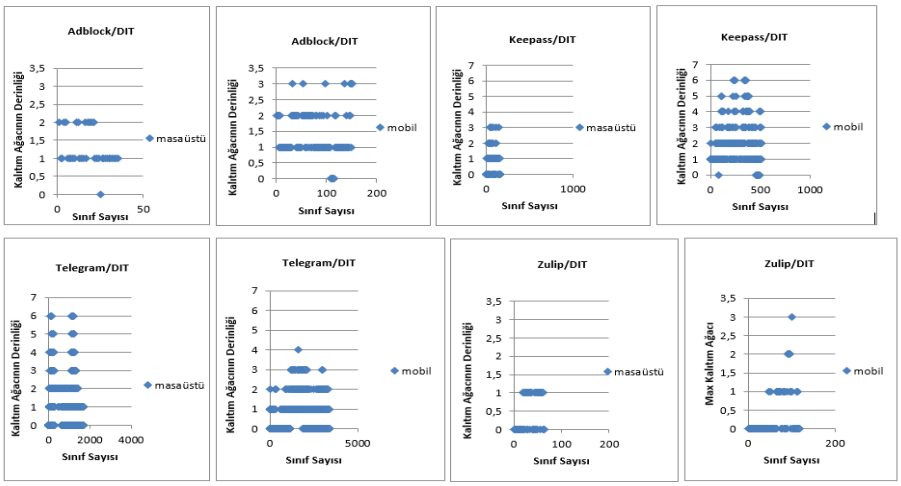
Sınıfın Tetiklediği Metot Sayısı (RFC) metriği incelendiğinde düşük değerde olması beklenen bu metrik değerinin KeePass ve Telegram uygulamalarının mobil versiyonlarında, AdBlock ve Zulip uygulamalarının ise masaüstü versiyonlarında daha düşük değerde olduğu görülmüştür (Şekil 3).

Kalıtım Ağacının Derinliği (DIT) metriğine göre karşılaştırma yapıldığında bu metriğin ortalama değerinin birbirine yakın sonuçlara sahip olduğu görülmüştür. Ayrıca Telegram dışındaki uygulamaların mobil versiyonlarında kalıtım ağacının derinliğinin de daha yüksek seviyelerde seyrettiği tespit edilmiştir. (Şekil 4).

Sınıf Başı Ağırlıklı Metot Sayısı (WMC) metriği incelendiğinde AdBlock uygulaması dışındaki diğer üç uygulamanın mobil versiyonunda masaüstü versiyonuna göre daha düşük değerler elde edildiği görülmüştür (Şekil 5).



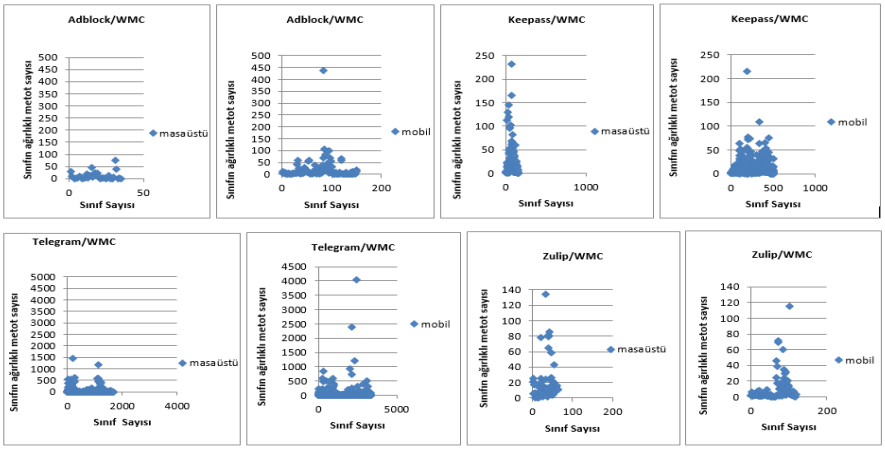
Şekil. 3. RFC Metrik Değeri vs Sınıf Sayısı



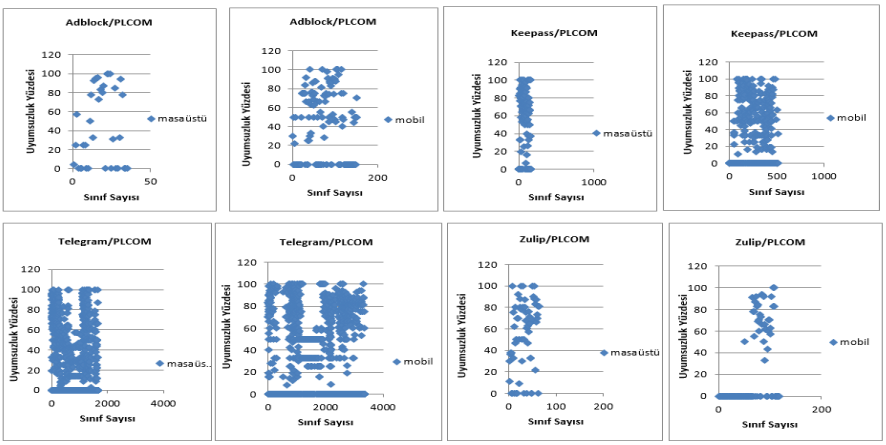
Şekil. 4. DIT Metrik Değeri vs. Sınıf Sayısı

Metot İçi Uyumsuzluk Yüzdesi (PLCOM) metriği incelendiğinde dört uygulamanın masaüstü ve mobil versiyonlarının uyumsuzluk yüzdesi karşılaştırıldığında tüm uygulamaların mobil versiyonlarında bu metrik değerlerinin masaüstü versiyonlarına oranla daha düşük olduğu gözlemlenmiştir. Bu nedenle bu metriğin mobil uygulamalarda etkin bir şekilde kullanılabileceği düşünülmektedir (Şekil 6).

Tablo 4'de sonuçların daha detaylı incelenebilmesi için yukarıdaki şekillerde grafiksel olarak gösterilen tüm parametreler birlikte verilmiştir.



Şekil. 5. WMC Metrik Değeri vs. Sınıf Sayısı



Şekil. 6. PLCOM Metrik Değeri vs. Sınıf Sayısı

6 Sonuç ve Gelecek Çalışmalar

Bu çalışmada, AdBlock, KeePass, Telegram ve Zulip açık kaynak kodlu uygulamalarının, masaüstü ve mobil versiyonları için CK metrik kümesi metrikleri "Understand" statik kod analiz aracı kullanılarak elde edilmiş ve elde edilen değerler karşılaştırılarak, geleneksel masaüstü uygulamalarında kullanılan yazılım metriklerinin mobil uygulamalarda etkin olarak kullanılabilirliği araştırılmıştır. Tablo 4 ve grafiklerden anlaşılacağı üzere CBO metriğinin davranışı Telegram – Zulip ve AdBlock - KeePass olarak sınıflandırabilir. Bu metrik Telegram – Zulip için mobil uygulamalarda daha yüksek değere sahipken AdBlock – KeePass için masaüstü uygulamalarda daha yüksek değere sahiptir. NOC metriği Telegram uygulaması hariç mobil uygulamalarda daha yüksek değer almıştır. RFC metriği ise uygulamaları davranış yönünden AdB-

lock – Zulip, KeePass – Telegram olarak gruplamıştır. AdBlock ve Zulip uygulamalarının mobil versiyonlarında RFC metriğinin daha yüksek değere sahip olduğu gözlenirken, KeePass ve Telegram uygulamalarının masaüstü versiyonlarında RFC değerinin daha yüksek değer aldığı görülmüştür. DIT metriğinin ortalama değerleri incelendiğinde AdBlock – KeePass uygulamalarında mobil versiyonların metrik değeri daha yüksekken, Telegram – Zulip uygulamalarında tam tersi bir durum söz konusudur. WMC metriği ise AdBlock hariç mobil versiyonlarda daha düşük değere sahiptir. PLCOM metriği tüm uygulamaların mobil versiyonlarında daha düşük değer almıştır.

Tablo 4. Metriklerin Uygulama Bazında Karşılaştırılması

Coupling Between Objects									
Project	Mean	SD	Max	Min	Project	Mean	SD	Max	Min
Adblock Masaüstü	2,31	2,30	11,00	0,00	Adblock Mobil	2,22	3,04	23,00	0,00
KeePass Masaüstü	5,00	6,27	32,00	0,00	KeePass Mobil	3,20	4,69	37,00	0,00
Telegram Masaüstü	4,40	14,80	515,00	0,00	Telegram Mobil	5,92	9,50	259,00	0,00
Zulip Masaüstü	4,09	3,54	17,00	0,00	Zulip Mobil	5,44	5,89	31,00	0,00

Number Of Children									
Project	Mean	SD	Max	Min	Project	Mean	SD	Max	Min
Adblock Masaüstü	0,20	0,90	5,00	0,00	Adblock Mobil	0,32	1,73	20,00	0,00
KeePass Masaüstü	0,19	0,71	4,00	0,00	KeePass Mobil	0,53	1,68	23,00	0,00
Telegram Masaüstü	0,64	10,16	336,00	0,00	Telegram Mobil	0,49	6,79	333,00	0,00
Zulip Masaüstü	0,03	0,25	2,00	0,00	Zulip Mobil	0,16	0,82	6,00	0,00

Response For A Class									
Project	Mean	SD	Max	Min	Project	Mean	SD	Max	Min
Adblock Masaüstü	5,46	7,74	38,00	1,00	Adblock Mobil	8,21	13,92	127,00	0,00
KeePass Masaüstü	14,84	15,34	88,00	0,00	KeePass Mobil	9,97	12,80	92,00	0,00
Telegram Masaüstü	37,68	57,81	451,00	0,00	Telegram Mobil	19,32	14,94	292,00	0,00
Zulip Masaüstü	10,13	9,85	43,00	0,00	Zulip Mobil	18,14	10,98	95,00	0,00

Max Depth Of Inheritance Tree									
Project	Mean	SD	Max	Min	Project	Mean	SD	Max	Min
Adblock Masaüstü	1,26	0,51	2,00	0,00	Adblock Mobil	1,33	0,64	3,00	0,00
KeePass Masaüstü	0,80	0,71	3,00	0,00	KeePass Mobil	1,93	1,22	6,00	0,00
Telegram Masaüstü	1,25	1,08	6,00	0,00	Telegram Mobil	0,55	0,82	4,00	0,00
Zulip Masaüstü	0,45	0,50	1,00	0,00	Zulip Mobil	0,31	0,61	3,00	0,00

Weighted Methods Per Class									
Project	Mean	SD	Max	Min	Project	Mean	SD	Max	Min
Adblock Masaüstü	11,14	16,11	77,00	1,00	Adblock Mobil	15,43	39,44	436,00	0,00
KeePass Masaüstü	21,52	33,56	232,00	0,00	KeePass Mobil	10,12	16,18	215,00	0,00
Telegram Masaüstü	19,33	69,71	1458,00	0,00	Telegram Mobil	13,24	93,07	4039,00	0,00
Zulip Masaüstü	18,83	25,21	134,00	0,00	Zulip Mobil	8,19	16,15	115,00	0,00

Percent Lack of Cohesion in Methods									
Project	Mean	SD	Max	Min	Project	Mean	SD	Max	Min
Adblock Masaüstü	43,57	40,45	100,00	0,00	Adblock Mobil	38,16	37,69	100,00	0,00
KeePass Masaüstü	49,53	37,77	100,00	0,00	KeePass Mobil	29,90	35,45	100,00	0,00
Telegram Masaüstü	22,68	31,83	100,00	0,00	Telegram Mobil	21,68	31,79	100,00	0,00
Zulip Masaüstü	54,25	33,24	100,00	0,00	Zulip Mobil	21,04	34,40	100,00	0,00

Sonuç olarak aynı uygulamanın masaüstü ve mobil sürümlerinin (içsel) kalite özellikleri arasında total bir korelasyon bulunamamıştır. Desktop uygulamasının kalitesinin mobil uygulamanın kalitesi üzerine direkt etkisi gösterilememiştir. Bu nedenle gelecek çalışmalarda metrik tabanlı analizlerin daha ayrıntılandırılması planlanmaktadır. Ayrıca gelecekte yapılması planlanan çalışmalar arasında, bu çalışmanın aynı programlama diliyle yazılmış masaüstü ve mobil uygulamalarda da uygulanması ve farklı metrikler ile kıyaslama yapılması bulunmaktadır.

7 Kaynaklar

1. Number of apps in leading app stores, 2016 Statistic, URL https://www.statista.com/ statistics/276623/number-of-apps-available-in-leading-app-stores.
2. ISO/IEC 9126-1:2001 - Software engineering -- Product quality -- Part 1: Quality model, URL http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=22749
3. Fenton, N. Bieman, J.: Software metrics: a rigorous and practical approach. CRC Press. (2014)
4. Sommerville, I.: Software Engineering, 9th ed., Pearson Education Inc. Addison-Wesley, Boston. (2011)
5. Taytaş, E.F., Gün, M., Dinçer, K., Baştüzel, S., Tekin, B.: Kamu Kurumları Tarafından Yazılım Satın Alma Sürecinde Kullanılacak Etkin Bir Yöntem Geliştirilmesi. Proceedings of the 9th Turkish National Software Engineering Symposium. Yaşar Üniversitesi, Izmir, Turkey. (2015)
6. Chidamber, S., Kemerer, C.: A metrics suite for object oriented design. IEEE Transactions on Software Engineering. 20, 476-493, (1994)
7. Abreu, F. B., Carapuça, R.: Object-Oriented Software Engineering : Measuring and Controlling the Development Process. 4th. International Conference of Software Quality, 4 (October), 3–5. (1994)
8. Bansiya, J., Davis, C. G.: A hierarchical model for object-oriented design quality assessment. IEEE Transactions on Software Engineering, 28(1), 4–17. (2002)
9. Mcquillan, J. A., Power, J. F.: Some observations on the application of software metrics to UML models - Position Paper. (2006)
10. AdBlockPlus, URL https://github.com/adblockplus
11. KeePass, URL https://github.com/keepassx
12. Telegram, URL https://github.com/telegram
13. Zulip, URL https://github.com/zulip
14. Jošt, G., Huber, J., Ičko, M. H. E. R.: Using Object Oriented Software Metrics for Mobile Application Development. Second Workshop on Software Quality Analysis, Monitoring, Improvement and Applications, 17–27. (2013)
15. Syer, M. D., Nagappan, M., Adams, B., Hassan, A. E.: Studying the relationship between source code quality and mobile platform dependence. Software Quality Journal, 23(3), 485–508. (2015)
16. Ural, E., Umut, T., Feza, B.: Nesneye Dayalı Yazılım Metrikleri ve Yazılım Kalitesi. Yazılım Kalitesi ve Yazılım Geliştirme Araçları Sempozyumu. (2008)
17. Marinescu, R.: Using object-oriented metrics for automatic design flaws detection in large scale systems. Lecture Notes in Computer Science, 1543, 252–252. (1998)
18. Deursen, A. Van, Magiel, B.: Predicting Class Testability using Object-Oriented Metrics. Fourth IEEE International Workshop. (2004)
19. Scitools, URL https://stage.scitools.com/
20. Scitools | Build Notes, URL https://scitools.com/build-notes/