Journal of the Faculty of Engineering and Architecture of Gazi University 34:4 (2019) 1807-1829

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University

Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

A two-dimensional method for evaluating maintainability and reliability of open source software

Nebi Yılmaz, Ayça Tarhan
Hacettepe Üniversitesi, Bilgisayar Mühendisliği Bölümü, Ankara, 06800, Türkiye

Highlights:

- Evaluation method for maintainability and reliability in Open Source Software
- Code-based and community-based evaluation
- Implementation of the proposed method for three Open Source Software products and a comparison of the results with other methods

Keywords:

- Open source
- Software selection
- Software quality
- Maintainability
- Reliability
- Software metrics

Article Info:

Research Article
Received: 02.03.2018
Accepted: 07.12.2018

DOI:

10.17341/gazimmfd.571563

Correspondence:

Author: Nebi Yilmaz
e-mail: yilmaz@cs.hacettepe.edu.tr
phone: +90 546 948 2202

Graphical/Tabular Abstract

Increased popularity of open source software (OSS) has led to a considerable proliferation of alternative software. However, this being the case, an evident lack of studies that would contribute to evaluation of OSS by organizations has turned the process of selecting the most suitable product into an appealing research problem. In this article, a method to evaluate reliability and maintainability of OSS products by using both code-based and community-based aspects is proposed.

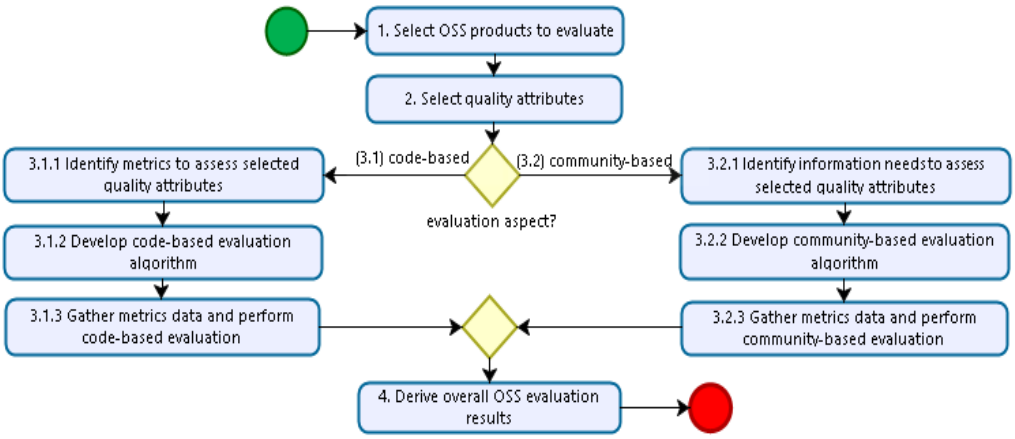


Figure A. Process followed to evaluate OSS products

Purpose: In this article, a method to evaluate reliability and maintainability of OSS by using both code-based and community-based aspects is proposed through the synthesis of existing studies in literature.

Theory and Methods: To perform code-based evaluation, some internal attributes of the most recent quality model, ISO/IEC 25010, are selected and object-oriented C&K metrics are employed in an attempt to measure these attributes. To perform community-based evaluation, metrics derived from historical data such as e-mailing lists, problem reports, frequently asked questions, and etc. are utilized to identify and satisfy information needs as conformant to ISO/IEC 15939 standard for software measurement process.

Results: The two-dimensional method was used in selection of the Java build tools written in Java, and the results obtained by applying the proposed method and the results obtained by using OSMM and OpenBRR which are common methods in the literature were compared. According to the evaluation and comparison results, the results obtained with the existing models confirm each other and the results obtained with the proposed method

Conclusion: The proposed method can be subject to further empirical studies on OSS selection with respect to maintainability and reliability attributes.

Journal of the Faculty of Engineering and Architecture of Gazi University 34:4 (2019) 1807-1829

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University

Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

Açık kaynak yazılımlarda bakım yapılabilirliği ve güvenilirliği ölçmek için iki boyutlu değerlendirme metodu

Nebi Yılmaz*, Ayça Tarhan
Hacettepe Üniversitesi, Bilgisayar Mühendisliği Bölümü, Ankara, 06800, Türkiye

ÖNEÇIKANLAR

- Açık Kaynak Yazılımlarda (AKY) bakım yapılabilirlik ve güvenilirlik için değerlendirme metodu
- Kod-tabanlı ve toplum-tabanlı değerlendirme
- Önerilen metodun üç AKY ürünü üzerinde uygulanması ve sonuçların diğer metotlarla karşılaştırılması

Makale Bilgileri
Araştırma Makalesi
Geliş: 02.03.2018
Kabul: 07.12.2018

DOI:
10.17341/gazimmfd.571563

Anahtar Kelimeler:
Açık kaynak, yazılım seçimi, yazılım kalitesi, bakım yapılabilirlik, güvenilirlik, yazılım metrikleri

ÖZET

Son yıllarda açık kaynak yazılımların (AKY) popülerliğinin artması, birbirine alternatif olarak pazara sunulan bu tür yazılımların sayısının hızla artmasına sebep olmuştur. Ne var ki açık kaynak yazılımlarının kalitesinin potansiyel kullanıcılar tarafından nasıl değerlendirilebileceğine ışık tutan akademik çalışmalar sınırlı sayıdadır. Bu makalede, literatürdeki mevcut çalışmalar sentezlenerek açık kaynak yazılımlarda bakım yapılabilirliği ve güvenilirliği ölçmek amacıyla önerilen metot anlatılmıştır. Bu ölçümle açık kaynak yazılımlar, hem kod-tabanlı hem de toplum-tabanlı olarak iki boyutlu bir metotla değerlendirilmektedir. Kod-tabanlı ölçüm için en güncel kalite modeli olan ISO/IEC 25010'un seçilen içsel öznitelikleri ve bu öznitelikleri ölçmek için nesneye yönelik C&K metrikleri kullanılmıştır. Toplum-tabanlı ölçüm için ise sistem ve yazılım mühendisliği için ölçüm süreci standardı olan ISO/IEC 15939 rehber alınarak bilgi ihtiyaçları belirlenmiş ve bu bilgi ihtiyaçlarını değerlendirmek için, ürünlerin veri tabanlarında depolanan elektronik posta listeleri, problem (hata) raporları, sıkça sorulan sorular vb. tarihsel verilerden türetilen metrikler kullanılmıştır. Bu çalışmada önerilen iki-boyutlu metot, Java dilinde yazılmış üç Java programı inşa aracının (Java build tool) seçiminde kullanılmış ve önerilen metot uygulanarak elde edilen sonuçlar ile literatürde yaygın bilinen metotlardan olan OSMM ve OpenBRR uygulanarak varılan sonuçlar karşılaştırılmıştır. Değerlendirme ve karşılaştırma sonuçlarına göre, mevcut modellerle elde edilen sonuçlar, hem birbirlerini hem de önerilen modelle elde edilen sonuçları doğrular niteliktedir.

A two-dimensional method for evaluating maintainability and reliability of open source software

HIGHLIGHTS

- Evaluation method for maintainability and reliability in Open Source Software
- Code-based and community-based evaluation
- Implementation of the proposed method for three Open Source Software products and a comparison of the results with other methods

Article Info
Research Article
Received: 02.03.2018
Accepted: 07.12.2018

DOI:
10.17341/gazimmfd.571563

Keywords:
Open source, software selection, software quality, maintainability, reliability, software metrics

ABSTRACT

Increased popularity of open source software (OSS) has led to a considerable proliferation of alternative software. However, there is a lack of studies in literature that shed light into the evaluation of OSS by organizations. In this article, a method to evaluate reliability and maintainability of OSS by using both code-based and community-based aspects is proposed through the synthesis of existing studies in literature. To perform code-based evaluation, some internal attributes of the most recent quality model, ISO/IEC 25010, are selected and object-oriented C&K metrics are employed in an attempt to measure these attributes. To perform community-based evaluation, metrics derived from historical data such as e-mailing lists, problem reports, frequently asked questions, and etc. are utilized to identify and satisfy information needs as conformant to ISO/IEC 15939 standard for software measurement process. The two-dimensional method was used in selection of the Java build tools written in Java, and the results obtained by applying the proposed method and the results obtained by using OSMM and OpenBRR which are common methods in the literature were compared. According to the evaluation and comparison results, the results obtained with the existing models confirm each other and the results obtained with the proposed method.

*Sorumlu Yazar/Corresponding Author: nebiyilmaz10@gmail.com, atarhan.hacettepe@gmail.com / Tel: +90 0312 297 7500

1. GİRİŞ (INTRODUCTION)

Açık kaynak yazılım (AKY); kaynak kodları özel bir telif hakkı lisansı ile herkesin incelemesine, kullanımına ve dağıtımına açılan, böylece kullanıcıya yazılımı değiştirme özgürlüğü sunan ve dünyanın her tarafından bilişim uzmanlarınca imece yöntemi ile endüstri standartlarında geliştirilen yazılımdır [1]. AKY'ların tercih edilmesinin sebepleri, geçmişten günümüze farklılık göstermektedir. Geçmiş yıllarda AKY kullanımının ana sebebi, mevcut kod tabanının ihtiyaca göre değiştirilerek tekrar kullanımının kaynak ve zaman tasarrufu sağlamasıydı. Son yıllarda ise AKY'ların yüksek kaliteli, güvenilir ve güvenli olarak algılanmaya başlaması, tercih edilmelerinde temel sebepler olmuştur. Böyle algılanmasındaki temel sebep ise AKY ürünlerinin birçok geliştiricinin dikkatli incelemesinden geçmiş ve dolayısıyla hatalardan arındırılmış olduğunun kabul edilmesidir.

Black and Duck Software [2] ve North Bridge Venture [3] raporlarına göre, özellikle 2010 yılından sonra AKY'ların kullanım miktarı dikkat çekmektedir. Noyes tarafından 2013 yılında yapılan AKY kullanımına yönelik anket çalışmasının sonuçları da AKY'ların kullanımının arttığını doğrular niteliktedir [4]. AKY popülerliğinin bu denli artması sonucunda, bu ürünleri sağlayan Github, SourceForge, Apache, Debian, Savannah vb. sitelerdeki proje ve kullanıcı sayıları da devasa şekilde büyümüştür. Sadece Github'ın 2013 yılından sonra aktif kullanıcı sayısı 10 milyonu geçmiştir [5]. Ayrıca SourceForge'un güncel verilerine baktığımızda, kullanıcılar tarafından haftada ortalama 32 milyon proje indirildiği gözlenmektedir [5]. AKY sayısı bu derece artınca ürünlerin alternatiflerinin sayısı da ilişkili olarak artmıştır. Kullanıcıların yazılımları benimsediği ve yanlış yazılım ürünü seçiminin şirketleri büyük maddi kayıplara maruz bıraktığı düşünüldüğünde, ihtiyacı karşılayan doğru ürünü seçmenin kritik hale geldiği görülebilir.

Hazır (kaynağı açık olmayan) yazılımlarda olduğu gibi AKY'larda da en büyük zorluk, değerlendirme ve seçme işlemidir [7]. Literatürde yazılım seçimine yönelik olarak çok-kriterli karar verme yöntemlerinin, genellikle hazır yazılımlar için kullanıldığı görülmektedir [21, 22]. Bu yöntemler belirli sayıda özellik üzerinden, nesnel (kural-odaklı) seçim yapılmasına imkân tanımakla birlikte, AKY seçiminde mevcut modellerde önemi vurgulanan kullanıcı ihtiyaçlarını çoğunlukla dikkate almamaktadır ve az sayıda çalışmada kullanılmıştır [23]. AKY'ların hazır yazılımlara kıyasla en temel özelliklerinden biri, kaynak kodlarının kullanıcıya açık olmasıdır ve literatürde bu kaynak kodları ölçmek için birçok metrik mevcuttur. Bununla birlikte, hangi özniteliğin (attribute) hangi metrikler kullanılarak nasıl ölçüleceği konusunda sınırlar ve kurallar net değildir. AKY'ların bir diğer temel özelliği ise bu ürünlerin piyasaya sürülmesinden itibaren, veri tabanlarında geçmişe dönük verilerinin tutulmasıdır. Bunlar; yazılım büyüklüğü [6] karşılaşılan hata sayıları, eklenen ve silinen satır sayıları, aktif geliştirici sayıları, hata önem dereceleri, geliştirici-geliştirici ve kullanıcı-geliştirici arasındaki eposta sayıları, hatanın durumu (örn. açık/kapalı/çözüldü) vb. verilerdir. Bu verilerin bolluğuyla birlikte, AKY'ların değerlendirilmesinde ihtiyaçların nasıl belirleneceği ve bu ihtiyaçlar doğrultusunda bahsedilen verilerin nasıl kullanılacağı, gerçekten karışık bir iştir. Hazır yazılımlar kadar olmasa da AKY değerlendirme ve seçimi için, akademik ve endüstriyel alanlarda bazı metotlar önerilmiştir [8-11]. Ne var ki standartlaşmış bir metot yoktur ve bu eksiklikten dolayı kullanıcılar, ihtiyaçlarını karşılayan AKY ürünlerini öznel değerlendirmeler veya tavsiyeler üzerine seçmektedir [12].

Yukarıda anlatılan zorluklardan hareketle bu çalışmada, kullanıcının hâlihazırda öznellikle gerçekleştirdiği AKY seçimini, kendi ihtiyaçlarının farkında olarak yapmasını ve aynı zamanda değerlendirmede objektif unsurları kullanmasını destekleyecek metrik tabanlı bir değerlendirme yöntemi önermek hedeflenmiştir. Bu amaca yönelik olarak, deneysel çalışmalardan soyutlama yoluyla metot tanımlamayı yönlendiren Tasarım Bilimi Araştırma (Design Science Research) [31] yönteminden faydanılmıştır. AKY'ları değerlendirmeyi ve seçmeyi yönlendirmek üzere deneysel örneklem bağlamında, *bakım-yapılabilirlik* ve *güvenilirlik* kalite özniteliklerine odaklanılmıştır. Bakım yapılabilirlik ve güvenilirlik, geçmişten günümüze bütün kalite modelleri tarafından kapsanmıştır ve güncel kalite modeli olan ISO/IEC 25010 [13] tarafından da ürün kalite ölçmeye temel alınmaktadır. Bu çalışmada önerilen metot ile AKY ürünü, bakım-yapılabilirlik ve güvenilirlik öznitelikleri için; hem *kod-tabanlı* hem de *toplum-tabanlı* açıdan, iki ayrı boyutta değerlendirilmekte ve boyutlara ilişkin değerlendirme sonuçları birleştirilerek ürüne ilişkin tek bir kalite indeks değeri elde edilmektedir. Önerilen metot, mevcut çalışmaların [8, 9-11],]25, 28] iyi yönlerini temel alarak tümleştirmektedir. Bu makalede iki boyutlu değerlendirme metodunun, bakım-yapılabilirlik ve güvenilirlik öznitelikleri bağlamında detayları ve metodun üç açık kaynak Java programı inşa aracının seçimine uygulanması ile varılan sonuçlar anlatılmıştır. Ayrıca uygulama sonuçları, literatürde AKY'ları değerlendirmek için önerilmiş, bilinen modellerin uygulama sonuçlarıyla karşılaştırılmıştır.

Bu makalenin kalanı şu şekilde düzenlenmiştir: İkinci bölümde ISO/IEC 25010 kalite modeli, ISO/IEC 15939 sistem ve yazılım mühendisliği için ölçüm standartı ve değerlendirmede kullanılan yazılım metrikleri ile ilgili önbilgi verilmiştir. Üçüncü kısımda ilişkili çalışmalar anlatılmış ve çalışmamızın bunlarla benzer ve farklı yönleri açıklanmıştır. Dördüncü bölümde önerilen metodun genel süreci ve bu sürecin bakım yapılabilirlik ve güvenilirlik kalite özniteliklerini değerlendirmek için uyarlanması anlatılmıştır. Beşinci bölümde, metodun uygulanması ile üç Java inşa aracının bakım yapılabilirlik ve güvenilirlik değerlendirmesine ilişkin detaylar ve sonuçlar verilmiştir. Altıncı bölümde metodun ölçüm sonuçlarının doğruluğunu

sınamak amacıyla elde edilen sonuçlar, literatürde AKY'ları değerlendirmeye yarayan, bilinen iki modelin sonuçları ile karşılaştırılmıştır. Son ve yedinci bölümde ise genel sonuçlar özetlenmiş, çalışmaya ve sonuçlara yönelik kısıtlar ile gelecek çalışmalardan bahsedilmiştir.

2. ÖN BİLGİ (BACKGROUND)

2.1. ISO/IEC 25010 Yazılım Kalite Modeli ve Öznitelikleri (ISO / IEC 25010 Software Quality Model and Attributes)

ISO/IEC 25010 kalite modeli [13] yazılım ürününün kalitesini, paydaşların gereksinimlerini karşılama derecesi olarak tanımlar. Model bu gereksinimleri belirlediği öznitelikler ve alt özniteliklerden oluşan hiyerarşik bir yapıyla tanımlar. Bu hiyerarşik modeli önerirken daha eski bir kalite modeli olan ISO/IEC 9126'yı [14] temel almıştır. Ürüne yönelik iç ve dış kaliteyi tanımlamak üzere, ISO/IEC 9126 kalite modelini güncelleyerek ve yeni özellikler ekleyerek Tablo 1'de gösterildiği gibi, sekiz ana öznitelikten oluşan yeni bir kalite modeli önermiştir. Bu çalışmada, ISO/IEC 25010'un güvenilirlik ve bakım yapılabilirlik öznitelikleri, kullanıcı ihtiyacına özel olarak AKY seçimi için temel alınmıştır. Ayrıca kullanıcının, aday AKY'ları kullanımdan önce değerlendireceği varsayıldığı için, ISO modelinin "kullanımdaki kalite" boyutu önerilen yönteme dâhil edilmemiş ve modelde öznitelikleri aynı olan iç ve dış kalite boyutlarına odaklanılmıştır.

2.2. ISO/IEC 15939 Yazılım Ölçme Süreci (ISO/IEC 15939 Software Measurement Process)

ISO/IEC 15939 sistem mühendisliği, yazılım mühendisliği ve yönetim disiplinleri için geçerli olan bir ölçüm sürecini tanımlar [15]. Süreç, hangi ölçüm bilgisinin gerekli olduğunu, ölçümlerin ve analiz sonuçlarının nasıl uygulanacağını ve analiz sonuçlarının geçerli olup olmadığını belirtir. Kalite bakımından ürünün değerlendirilmesinde, ilgili verilerin toplanmasında ve sonuçların ihtiyaca uygun değerlendirilmesinde yol göstericidir. Süreç, Planla-Uygula-Denetle-Harekete geç döngüsüyle uyumlu, dört ana aşamadan oluşur: 1) *Yükümlülükleri belirle*: Ölçmenin amacı tanımlanır ve yönetim isteklerine göre sorumluluklar tespit edilir; daha sonra da ölçmeye ilişkin görevler ve kaynaklar atanır. 2) *Ölçme sürecini planla*: Bilgi ihtiyaçları belirlenir ve bu ihtiyaçlar öneme göre sıralanır; hangi metriklerin kullanılacağı belirlenir ve verilerin toplanacağı araçlar sağlanır. 3) *Ölçme sürecini gerçekleştir*: İhtiyaçlar doğrultusunda veriler toplanır ve toplanan veriler doğrulanır; verilerin analizi yapılarak analiz sonuçları belgelendirilir ve paydaşlara iletilir. *4) Sonuçları değerlendir*: Ölçme süreci genel olarak değerlendirilir. ISO/IEC 15939 ölçme sürecindeki temel kavramların tanımları Tablo 2'de verilmiştir.

2.3. Nesneye Yönelik Metrikler (Object-oriented metrics)

Yazılım dünyasında nesneye yönelik geliştirilen programların ölçümünü yapmak ve bireysel değerlendirmelerden bağımsız kalabilmek için birçok metrik önerilmiştir. Bunlardan en önemlisi ve en çok kullanılanı Shyam R. Chidamber ve Chris F. Kemerer tarafından önerilmiş olan C&K metrik setidir [16]. Bu çalışmada kod-tabanlı boyutta değerlendirmeye temel alınan C&K metrikleri (CBO, DIT, NOC, RFC, WMC ve LCOM) ile ek olarak kullanılan (CC, NOS ve NLL) metrikleri, Tablo 3'te açıklanmıştır.

3. İLİŞKİLİ ÇALIŞMALAR (RELATED WORKS)

AKY ürünleri yazılımın kaynak koduyla birlikte kullanıcılara birçok bilgi sunduğu için değerlendirme aşamasında bu bilgilerin kullanılması ve bu bilgiler ışığında en iyi ürünün seçilmesi zor bir süreçtir. Bu yüzden mevcut çalışmalar bu bilgileri kullanmada farklı yollar izlemiş ve farklı metotlar ve modeller önermişlerdir [19].

OSMM (Open Source Maturity Model) [8], bir AKY ürününün organizasyon için uygun olup olmadığını kontrol eden modeldir. Ürünlerin olgunluğunu (maturity) temel

Tablo 1. ISO/IEC 25010 kalite modeli öznitelikleri ve alt-öznitelikleri [13]
(ISO/IEC 25010 quality model attributes and sub-attributes)

Öznitelikler	Alt öznitelikler
Fonksiyonel uygunluk	Uygunluk, doğruluk, birlikte işlerlik, güvenlik, uyumluluk
Güvenilirlik	Olgunluk, hata dayanıklılığı, geri kazanılabilirlik, uyumluluk
Performans verimliliği	Zamana göre davranış durumu, kaynak kullanımı, uyumluluk
İşletilebilirlik	Uygunluk, tanınabilirlik, kullanım kolaylığı, öğrenilebilirlik, çekicilik, teknik ulaşılabilirlik, uyumluluk
Güvenlik	Gizlilik, bütünlük, inkâr edilememe, hesap verebilirlik, aslına uygunluk, uyumluluk
Uyumluluk	Değiştirilebilirlik, birlikte var olabilme, birlikte işlerlik, uyumluluk
Bakım-yapılabilirlik	Modülerlik, tekrar kullanılabilirlik, çözümlenebilirlik, değişebilirlik, değişim kararlılığı, sınanabilirlik, uyumluluk
Aktarılabilirlik	Taşınabilirlik, adapte olabilirlik, yüklenebilirlik, uyumluluk

Tablo 2. ISO/IEC 15939 ölçme sürecindeki temel kavramlar (Basic concepts of ISO/IEC 15939 measurement process)

Kavram	Tanımı
Varlık (Entity)	Niteliklerin ölçülmesiyle tanımlanan nesnelerdir. Tipik yazılım mühendisliği nesneleri; ürünler, süreçler, projeler ve kaynaklardır.
Öznitelik	Bir varlığın, niceliksel ya da nitel olarak insan ya da otomatik araçlarla ayırt edilebilen karakteristiğidir.
Temel Metrik (Base Measure)	Bir nitelik ve onu nicelleştirme yöntemi açısından tanımlanan bir ölçüdür. Temel metrik, diğer metriklerden fonksiyonel olarak bağımsızdır ve tek bir öznitelik hakkında bilgi içerir.
Ölçme Yöntemi (Measurement Method)	Bir özniteliğin belirli bir ölçeğe göre sayısallaştırılmasında kullanılan mantıksal işlemler dizisidir. Ölçme yönteminin türü, öznel (Subjective) ve nesnel (Objective) olarak ikiye ayrılır.
Türetilmiş Metrik (Derived Measure)	İki ya da daha fazla temel metriğin bir fonksiyonu olarak tanımlanan bir ölçüdür.
Ölçme Fonksiyonu (Measurement Function)	İki ya da daha fazla temel metriği birleştirmek için gerçekleştirilen bir algoritma veya hesaplamadır.
Gösterge (Indicator)	Tanımlanmış bilgi ihtiyaçlarına göre bir modelden türetilen, belirli niteliklerin tahmin ya da değerlendirilmesini sağlayan bir öğedir. Göstergeler, analiz ve karar verme için temel oluşturur.
Model	Bir ya da daha fazla temel ve/veya türetilmiş metriği ilişkili karar kriterleriyle birleştiren algoritma ya da hesaplamadır. Modeller, tanımlanan bilgi ihtiyaçlarıyla ilgili tahminler veya değerlendirmeler üretir.
Bilgi Ürünü (Information Product)	Bilgi ihtiyaçlarını karşılayan ölçme sürecinin nihai ürünüdür.

Tablo 3. Nesneye yönelik metrikler ve tanımları (Object-oriented metrics and definitions)

Metrik	Tanımı
Nesne Sınıfları Arasındaki Bağımlılık (CBO)	Sınıfın bağlı olduğu sınıf sayısını ölçer. Bu bağımlılık sınıf içerisindeki bazı özelliklerin veya metotların başka sınıflarda, sınıflar arasında kalıtım olmaksızın kullanılması durumundaki bağımlılıktır [16]. Sınıflar arasındaki bağımlılığın fazla olması modüler tasarıma zarar verir [17] ve değişebilirliği azaltır.
Kalıtım Ağacının Derinliği (DIT)	Sınıfın kalıtım ağacının köküne uzaklığını ölçer [17]. Ağaç derinliğinin fazla olması daha fazla sınıf ve metot içereceği için karmaşıklığı arttırır ve yazılımın değişebilirliğinin ve ürünün kararlılığının düşük olduğunu gösterir.
Alt Sınıf Sayısı (NOC)	Bir sınıftan türetilmiş alt sınıfların sayısını ölçer [16]. Bu metrik değerinin fazla olması, yeniden kullanımının yüksek olduğunu, daha çok hatanın oluşabileceğini [16] ve test yapılırken harcanan çabanın yüksek olacağını gösterir [17].
Sınıfın Tetiklediği Metot Sayısı (RFC)	Bir sınıftan bir nesnenin metotlarının çağırılması durumunda, bu nesnenin tetikleyebileceği tüm metotların sayısıdır. Yani, bir sınıfta yazılan ve çağrılan toplam metot sayısıdır [17]. RFC metriğinin değeri düşük olduğunda yazılım ürünleri daha anlaşılır ve sınanabilirdir.
Sınıfın Ağırlıklı Metot Sayısı (WMC)	Bir sınıftaki metotların karmaşıklık derecesi ve sayısıdır [18]. Metotların sayısı arttıkça kodun çözümlenebilirlik süresi de otomatik olarak artacaktır.
Metotların Uyum Eksikliği (LCOM)	Metotların birbiriyle benzerlik derecesini ölçer [18]. Metrik değeri ne kadar düşükse değişebilirlik daha yüksektir.
Siklomatik Karmaşıklık (CC)	Program kaynak kodu akışının birbirinden bağımsız yolları takip etme oranını ölçer ve doğrudan kodun karmaşıklığı ile ilgilidir. Bu metrik değerinin büyük olması istenmeyen bir durumdur ve kaynak kodun çözümlenebilirliğine etki eder [9].
İç İçe Döngü Sayısı (NNL)	Bir sınıf içerisindeki döngülerin iç içe geçme derinliğini ölçer ve ne kadar büyük bir değer alırsa kodun sınanabilirliği ve kararlılığı azalır [9].
Açıklamaların Sayısı (NOS)	Program içerisindeki karmaşıklığı azaltmak adına, bize yol gösterecek yorumların ve açıklamaların sıklığını ölçer. Programın takip edilmesini ve çözümlenebilmesini kolaylaştırır [9].

alarak ve iki ayrı indikatör kullanarak değerlendirmeyi yapar: Ürün (product) indikatörü ve uygulama (application) indikatörü. Ürün indikatörü değerlendirilen AKY ürününün gelecekteki davranışı ile ilgili yapılan öngörüsel sonuçların göstergesidir. Uygulama indikatörü ise AKY ürününün kullanımı ile elde edilen ölçüm sonuçlarının göstergesidir.

OSMM, ürün indikatöründe 12 ve uygulama indikatöründe 15 olmak üzere 27 tane alt indikatörden oluşur. Kullanıcı bu alt indikatörlere 1, 3 ve 5 olmak üzere skorlar vererek her iki açıdan en iyi ürünü belirler. OSMM, bu indikatörlerin hiyerarşik listesini ve skorlar verilirken uyulması gereken kriterleri tanımlamıştır. Ancak bu indikatörler ürünün kaynak kodunu değerlendirme imkânı sunmadığı için, model bu yönüyle eleştiri almış ve yetersiz görülmüştür [9].

OpenBRR (Open Business Readiness Rating) [10], AKY'ların değerlendirilmesi için önerilmiş diğer bir modeldir. Değerlendirme için genellikle üst-düzey (high-level) metrikler temel alınmıştır. Bu metrikler arasında işlevsellik, operasyonel yazılım özellikleri, destek, servis, benimseme (adoption), geliştirme süreci sayılabilir. OpenBRR, bu metrikleri hiyerarşik olarak alt metriklere ayırır ve bunlara ağırlıklar vererek değerlendirme yapılır. Bu metrikler (örn. işlevsellik) öznel değerlendirmeler yapmaya yatkındır. Bu sebeple bu model, değerlendirmelerin öznelliği (kullanıcı-odaklı) yönünün fazla olmasından dolayı eleştiri almıştır [9].

Bu alanda önerilen diğer bir model QSOS (Qualification and Selection of Open Source Software) [11] modelidir ve iterasyonlarla ilerleyen dört aşamadan oluşur. Birinci aşamada değerlendirme yapılacak kriterler tanımlanır. İkinci aşamada değerlendirilecek AKY ürünü için belirlenen kriterler dâhilinde bilgiler toplanır ve kriterlere 0-2 arasında skor verilir (hangi değer aralığı için hangi skorun atanacağı metotta belirtilmiştir). Üçüncü aşamada, belirlenen kriterlere göre filtreler tanımlanır ve bu filtreler sayesinde ilişkisiz ürünler elenir. Son aşamada ise kullanıcı ihtiyaçlarını karşılayan uygun AKY'ler belirlenir. QSOS hiyerarşisindeki kriterler objektif değerlendirmeye uygun olduğu ve kullanıcıya bağımlı olmadığı halde, bu metodun yeterince esnek olmaması ve uygulanmasının zor olması bir dezavantajdır [9]. Ayrıca bu metodun skor değerlerinin 0 ile 2 arasında olması eleştirilmiş ve ürün seçimlerinde aralığın küçük olmasının kullanıcının işini zorlaştırdığı berlirtilmiştir [9]. Diğer güncel modellerden biri, QOS-OSS (Software Quality Observatory for Open Source Software) modelidir [9]. Bu model bizim önerimizde olduğu gibi değerlendirmesini yaparken hem kaynak koddan hem de ürünün bize sağladığı tarihi verilerden faydalanmıştır. Ölçüm için hiyerarşik yapıdan oluşan kalite modeli tanımlamıştır. Değerlendirme yaparken otomatik değerlendirme süreci ön planda ve kullanıcının görüşleri ikinci planda kalmıştır. Ürün seçimi için kullanıcı ihtiyaçlarının önemli olduğu düşünüldüğünde metodun bu ihtiyacı tam karşılayamaması olasıdır. Ayrıca toplum-tabanlı metriklerin otomatik değerlendirmeye yatkın metrikler olarak seçilmeye çalışılması, değerlendirmede kullanılabilecek metriklere bir sınırlama getirmektedir. Yukarıda anlattığımız, bizim çalışmamıza yön vermeyi sağlayan metotların yanında, literatürde 20'den fazla model mevcuttur. Yer kısıtı sebebiyle bu modellerin tümüne yer verilememiştir. Örnek olarak, Stoll ve Babar [20] literatürde bulunan 20 tane değerlendirme metodunu ele almış ve çalışmalarında, AKY seçiminde hangi modelin kullanıcı için uygun olduğunu anlamak amacıyla FOCOSEM (Framework for Comparing Open Source Software Evaluation Methods) [20] isminde bir çerçeve (framework) geliştirmişlerdir. Bu çerçeve, değerlendirme modeli seçimine yönelik birçok sorudan oluşmaktadır. Kullanıcı bu soruların cevaplarına göre kendine en uygun değerlendirme modelini seçmektedir.

Bu çalışmada yukarıda anlatılan AKY'ların değerlendirilmesine yönelik metotların iyi yönleri alınarak kapsamı kullanıcı ihtiyacına göre şekillenebilen bir metot geliştirilmek istenmiştir. Bizim çalışmamızın önceki çalışmalara benzerlikleri Tablo 4'te özetlenmiştir. Tabloda gösterildiği gibi, bazı metotlarda kullanıcı değerlendirmesi (kullanıcı-odağı) ön plana çıkarılmış bazılarında ise değerlendirme tamamen otomatik (kural-odaklı) yapılmıştır. Bazı çalışmalarda sadece toplum-tabanlı değerlendirmeye ağırlık verilmiş (örn. OSMM) bazılarında ise kod-tabanlı değerlendirme önemsenmiştir (örn. QOS-OSS). Bizim çalışmamızda, bilgi ihtiyacının belirlenmesinde değerlendirici ihtiyaçlarının ön plana çıkarıldığı (kullanıcı-odaklı) ama ölçüm sonuçlarına skor verirken değerlendiriciden bağımsız (kural-odaklı) bir metot sunulmuştur. Bu metot öznel olarak belirlenen bilgi ihtiyaçlarının nesnel olarak ölçülmesine olanak sağlar. Literatürdeki çalışmalar çoğunlukla ürünlerin genel kalitesini ölçmeye yöneliktir; önerilen yöntem ise kullanıcının ihtiyacına göre belirlediği kalite öznitelikleri bazında şekillenebilir. Ayrıca diğer metotlar, AKY'ların karakteristiğini yansıttığını düşünerek önerdikleri yeni kalite modelleri çerçevesinde ölçümlerini yapmışlardır. Bizim önerdiğimiz metotta ise uluslararası standart olan ISO/IEC 25010 kalite modelinin öznitelikleri ve alt özniteliklerine bağlı kalınarak ölçümler yapılmıştır.

Tablo 4'te herhangi bir modele bağlı kalmadan yapılan deneysel (empirical) çalışmalar da verilmiştir. Wheeler çalışmasında [25], AKY değerlendirme sürecini ticari yazılımları değerlendirme süreci ile kıyaslamış ve AKY'ları toplum-tabanlı olarak dört aşamada değerlendiren bir süreç önermiştir. Sung vd. [26], AKY seçimi için ISO 9126 kalite modelini yeniden düzenleyerek yeni bir kalite modeli sunmuşlardır. Hanefi vd. [18], AKY'ların bazı özniteliklerini değerlendirirken metriklerin elde edilmesi konusunda yol göstermiş ve C&K metrik setini kullanmıştır. Bu metrikler verimlilik, karmaşıklık, anlaşılırlık, yeniden kullanılabilirlik, test edilebilirlik ve dayanıklılık ile ilişkilendirilerek bu özniteliklerin ölçümünde kullanılmıştır. Garousi çalışmasında [27], AKY'ların başarısını ölçmek için dört AKY ürününü, üç toplum-tabanlı metrik kullanılarak karşılaştırılmıştır. Bu çalışma metriklerin kullanımı açısından bilgi sağlasa da ürünlerin kaynak kodlarını değerlendirmemektedir. Toplum-tabanlı değerlendirme tarafında Mackus vd. [28], Apache ve Mozilla'da bulunan tarihi verilerin neler olduğu, neler içerdiği ve nasıl kullanılması gerektiği hakkında kapsamlı bir inceleme yapmıştır. Bu çalışmada, sadece kod-tabanlı değerlendirme yapan çalışmalardan [9, 18, 24], [29, 30] ve sadece toplum-tabanlı değerlendirme yapan çalışmalardan [8, 10, 11], [25,

27, 28] farklı olarak, [9] çalışmasında yapıldığı gibi, iki boyutta değerlendirme yapılmıştır. Değerlendirme yapılırken bazı aşamalarda (örn. ihtiyaçların belirlenmesi), [8, 10] çalışmalarında olduğu gibi kullanıcı görüşleri önemsenerek (kullanıcı-odaklı), bazı aşamalarda (örn. ağırlıklandırma) ise [9, 11, 18], [24, 29, 30] çalışmalarında olduğu gibi kullanıcı görüşü önemsenmeden (kural-odaklı) değerlendirme yapılmıştır. Değerlendirme için [8-11, 26] çalışmalarında yapıldığı gibi yeni bir değerlendirme modeli önerilmemiş olup ISO/IEC 25010 kalite modeline bağlı kalınmıştır.

4. İKİ BOYUTLU DEĞERLENDİRME METODU (TWO-DIMENSIONAL EVALUATION METHOD)

AKY'ların değerlendirilmesi ve kullanıcı ihtiyacını karşılayan en uygun ürünün seçilmesi için geliştirilen metodun aşamaları Şekil 1'de verilmiştir. Bu çalışma tasarlanırken Information System (IS) araştırmaları için öngörülmüş [31] ve yazılım mühendisliği araştırmaları için de kullanılan [32, 33] Design Science Research (DSR) yöntemi gözetilmiştir. Şekil 1'de verilen yöntem, DSR esaslarına göre deneysel araştırma bağlamının bir sonucu olarak ve ondan soyutlanarak tanımlanmıştır. Bu çalışmada, kullanıcı ihtiyacına özel olarak bakım yapılabilirlik ve güvenilirliğin değerlendirilmesinde, şekilde özetlenen değerlendirme süreci izlenmiştir. Genel değerlendirme süreci takip edilerek bu iki öznitelikten farklı özniteliklerin de değerlendirilmesi mümkün olabilir.

İzleyen alt bölümlerde, Şekil 1'de verilen metodun güvenilirlik ve bakım yapılabilirlik özniteliklerinin değerlendirilmesinde kullanılmak üzere uyarlanan adımları açıklanmıştır.

Tablo 4. Literatürde AKY seçimi için önerilen metotlar ve deneysel çalışmaların karşılaştırılması
(Comparing the methods and experimental studies proposed for the selection of OSS in the literature)

	Çalışmalar	Değerlendirme boyutu		Değerlendirme şekli		Değerlendirme modeli	
		Kod-tabanlı	Toplum-tabanlı	Kullanıcı-odaklı (öznel)	Kural-odaklı (nesnel)	Yeni değerlendirme modeli	Mevcut modeli uyarlama
Metot öneren çalışmalar	OSMM [8]		X	X		X	
	OpenBRR [10]		X	X		X	
	QSOS [11]		X		X	X	
	QOS-OSS [9]	X	X		X	X	
	Önerilen metot	X	X	X	X		X
Deneysel çalışmalar	Stamelos vd. [24]	X			X		
	Wheeler [25]		X				
	Sung vd. [26]					X	
	Hanefi vd. [18]	X			X		
	Garousi [27]		X				
	Mackus vd. [28]		X				
	Antony vd. [29]	X			X		
	Antony [30]	X			X		

Şekil 1. AKY'ların kalite değerlendirmesi için geliştirilen metodun aşamaları
(Steps of method developed for quality evaluation of OSS)

4.1. Değerlendirmesi Yapılacak AKY'ların Seçilmesi (Adım 1) (Select OSS to evaluate (Step 1))

İlk adımda ihtiyaç sahipleri öncelikle, hangi çeşit yazılım ürünü için değerlendirme yapacaklarını belirlemelidir. Bu doğrultuda piyasada var olan, gereksinimlerini karşılayacak aday yazılım ürünleri araştırılmalıdır.

Çalışmada belirlenen deneysel örneklem bağlamında, kullanıcının eklenti geliştirmeye uygunluğu değerlendirme ihtiyacından hareketle, Java tabanlı kod inşa araçlarının (İng: Java build tool) incelenmesi hedeflenmiştir. Buna göre ilk önce İnternet arama motorları (Google, Yandex, Yahoo vb.) kullanılarak piyasada var olan, Java dilinde yazılmış kod inşa araçları araştırılmıştır. Ürünlerin işlevsellik yönünden birbirine alternatif olmalarına dikkat edilmiş ve üç tane AKY ürünü seçilmiştir: Apache Ant, Apache Maven ve Apache Archiva. Daha sonra AKY ürünleri sağlayan İnternet sitelerine (Apache, SourceForge, Debian, Savannah vb.) bakılarak seçilen ürünlerin popülerliği ve işlevsel açıdan birbirine alternatif olabilme dereceleri doğrulanmıştır. Seçilen bu ürünler; proje geliştirilirken proje içinde standart oluşturmak, geliştirme sürecini basitleştirmek, dokümantasyonu etkili bir şekilde oluşturmak, projelerde ihtiyaç duyulan kütüphanelere otomatik olarak erişilebilmek ve tümleşik geliştirme ortamına bağımlılığı ortadan kaldırmak gibi ortak özelliklere sahiptir. Bu ürünlerin aynı amaca hizmet etmesinin yanı sıra, çalışmada homojenlik faktörünü desteklemek üzere, tümünün Java dilinde yazılmış araçlar olmaları da seçimde önemli yere sahiptir. Tablo 5'te seçilen AKY ürünlerinin bilgileri verilmiştir.

4.2. Değerlendirme Yapılacak Kalite Özniteliğinin Belirlenmesi (Adım 2)
(Determine quality attribute to evaluate (Step 2))

İkinci adımda, AKY ürün alternatiflerinin hangi kalite özniteliklerine göre değerlendirileceği ve karşılaştırılacağı belirlenir. Geçmişten günümüze önerilen kalite modellerinden en bilinir olanları ve içerdikleri öznitelikler Tablo 6'da gösterilmiştir. Bazıları modeller öncekilere yeni öznitelikler eklemiş, bazıları önceki modellerin özniteliklerini modellerinden çıkarmış ve bazıları da başka model için önerilen bir özniteliği kendi modelinde bir özniteliğin alt özniteliği olarak kullanmıştır. Tablo 6'dan, bilinen modellerin önerdiği ortak özniteliklerin; verimlilik, güvenilirlik, bakım yapılabilirlik, taşınabilirlik ve kullanılabilirlik olduğu görülmektedir [11]. Ayrıca bir

Tablo 5. Değerlendirilecek aday AKY ürünlerinin özellikleri (Characteristics of the candidate OSS products to evaluate)

Ürün özellikleri	Apache Archiva	Apache Maven	Apache Ant
İnternet sitesi	http://archiva .apache.org/index.cgi	http://maven .apache.org/	http://ant.apache.org
Ürün türü	Kod inşa aracı	Kod inşa aracı	Kod inşa aracı
Programlama dili	JAVA	JAVA	JAVA
Üretim yılı	2006	2002	2000
Mail arşivi	http://archiva. apache.org/mail-lists.html	http://maven.apache.org/mail-lists.html	http://ant.apache .org/mail.html
Problem veri tabanı	http://issues.apache.org/ jira/browse/MRM	https://issues.apache.org /jira /browse/MNG	https://bz.apache.org/bugzilla/buglist. cgi?product=Ant

Tablo 6. En bilinir kalite modelleri ve öznitelikleri (Most known quality models and attributes)

Kalite öznitelikleri	Boehm	McCall	FURPS	ISO 9126	Dromey
Verimlilik	X	X	X	X	X
Güvenilirlik	X	X	X	X	X
Fonksiyonellik			X	X	X
Bakım yapılabilirlik	X	X	X	X	X
Taşınabilirlik	X	X		X	X
Kullanılabilirlik		X	X	X	X
Test edilebilirlik	X	X		X	
Doğruluk		X			
Anlaşılabilirlik	X			X	
Esneklik		X	X		
İnsan yönetimi	X				
Bütünlük		X		X	
Birlikte işlerlik		X		X	
Olgunluk					X
Değişebilirlik	X				
Tekrar kullanılabilirlik		X			X

literatür analizine göre [34], yazılım ürünleri seçilirken en çok dikkat edilen özellikler; bakım yapılabilirliğinin kolay olması, verimliliğinin yüksek olması, güvenilir olması, kolay taşınabilir olması ve kolay kullanılabilir olmasıdır. Bu çalışmadaki deneysel örneklem bağlamında belirlenen, kullanıcının Java tabanlı kod inşa araçlarının eklenti geliştirmeye uygunluğunu değerlendirme amacına yönelik olarak, güvenilirlik ve bakım yapılabilirlik kalite öznitelikleri seçilmiştir. Değerlendirilecek ürünlerin kod inşa aracı olması ve çalıştırıldıkları ortamlarda verimliliğe (kaynakların en iyi kullanımına) yönelik kaygının genelde düşük olması da, bu iki özniteliğin seçilmesinde etkili olmuştur. Ayrıca bir kod inşa aracının tipik olarak güvenilir olması ve aynı zamanda uzun ömürlü kullanımı açısından bakım yapılabilirliğinin kolay olması beklenir. Bu sebeplerle, seçilen üç Java tabanlı kod inşa aracı, bu iki öznitelik bakımından değerlendirilmiştir.

4.3. Değerlendirme Yapılacak Boyutun Seçilmesi (Adım 3) (Determine the aspect of evaluation (Step 3))

Bu çalışmada, kod-tabanlı ve toplum-tabanlı boyutların her ikisi de değerlendirme için dikkate alınmıştır.

4.3.1. Kod-tabanlı değerlendirme (Code-based evaluation)

Bu aşamada AKY'lar sağlayan sitelerden ürünlerin kaynak kodlarına erişilmiştir.

4.3.1.1. Seçilen kalite özniteliğini ölçmek için metriklerin belirlenmesi (Adım 3.1.1.) (Determine the metrics to measure selected quality attribute (Step 3.1.1.))

Bu aşamada, bir önceki aşamada (Adım-2) belirlenen öznitelikleri ölçmek için hangi metrikleri kullanabileceğimizin araştırılması gerekir.

Bu çalışmada, bakım yapılabilirliği ve güvenilirliği ölçmek için metriklerin belirlenmesi adımında, Tablo 7'de gösterildiği gibi, her bir öznitelik için araştırma soruları oluşturulmuş ve bu soruları çerçevesinde iki özniteliği ölçmeye yönelik metrikler araştırılmıştır. Bakım yapılabilirliği ve güvenilirliği değerlendirmede kullanılacak metrikleri belirlemek için ilk olarak, ISO/IEC 25010 standardı içindeki metrikler gözden geçirilmiştir. Ne var ki bu metrikler, nesneye yönelik özellikleri olan Java inşa araçlarını değerlendirmek için uygun görülmemiştir. Standartta tanımlanan metrikler içinden seçim yapılamamış olmasının önemli bir sebebi de tanımlanan metriklerin çoğunun geliştirme bağlamına ilişkin bilgi toplanmasını gerektiriyor olmasıdır. AKY'lar dünya üzerinde farklı konumlarda ve ortamlardaki farklı onlarca geliştirici tarafından üretildiklerinden, bağlamla ilgili bilgilere ulaşmak maalesef mümkün değildir. Bu sebeple literatürdeki çalışmalar araştırılmış ve bu iki özniteliği ölçmek için de Tablo 7'de listelenen metriklerin kullanılmasına karar verilmiştir. Bu metrikler tabloda gösterildiği gibi iki öznitelik ile ayrı ayrı ilişkilendirilmiştir. Metrikler belirlendikten sonra üç aday AKY ürününün kaynak kodlarının metrik değerlerini hesaplamak için bir kod analiz aracı olan Understand Scitool [42] kullanılmıştır.

4.3.1.2. Kod-tabanlı değerlendirme yapmak için adımların belirlenmesi (Adım 3.1.2.) (Determine the steps to perform code-based evaluation (Step 3.1.2.))

Bu aşamada her bir özniteliğin nasıl değerlendirileceğini ilişkin adımlar birbirinden farklı şekilde belirlenmiştir.

Bakım yapılabilirlik için: Bakım yapılabilirliği ölçmek için, bu özniteliğin alt öznitelikleri kullanılmış ve ilişkili çalışmalar gözetilerek belirlenen metrikler, Tablo 7'de gösterildiği gibi, bu alt özniteliklerle ilişkilendirilmiştir. Kod analiz aracı kullanılarak metrik değerleri elde edildikten sonra, bakım yapılabilirliğin ölçülebilmesi için adımlar belirlenmiştir. Metriklerin sayısal değerleri ve tanımları kullanılarak bakım yapılabilirliğin her bir alt özniteliği için, seçilen ürünlere ağırlıklar verilmiştir. Ayrıca kullanıcı da ihtiyacı doğrultusunda ve önem derecesine göre, bu alt özniteliklere ağırlıklar vermiştir.

Güvenilirlik için: Güvenilirliğin değerlendirilmesi için bakım yapılabilirlikle aynı metrik değerleri kullanılsa da farklı adımlar izlenmiştir. Literatürdeki çalışmalar araştırıldığında, C&K metrik seti ile güvenilirliğin direkt olarak ölçüdüğü ve güvenilirlikle C&K metrikleri arasında

Tablo 7. Bakım yapılabilirlik ve güvenilirliği ölçmek için kullanılan metrikler
(Metrics used to measure maintainability and reliability)

Araştırma Sorusu (AS)	AS-1: Hangi metrikler bakım yapılabilirliği (veya bakım yapılabilirliğin alt özniteliklerini) ölçmek için kullanılabilir?				AS-2: Hangi metrikler güvenilirliği (veya güvenilirliğin alt özniteliklerini) ölçmek için kullanılabilir?	
Metrik	Test edilebilirlik	Kararlılık	Değişebilirlik	Çözümlenebilirlik		
	NNL	CBO	CBO	CC	CBO	NOC
	NOC	DIT	DIT	NOS	DIT	RFC
	RFC	NNL	LCOM	WMC	LCOM	WMC
İlişkili Çalışmalar	[9, 16, 18], [35-39]				[29, 30], [40, 41]	

ters orantılı bir ilişki olduğu görülmüştür [43-45]. Başka bazı çalışmalar [29, 30, 40], literatürde mevcut olan önceki çalışmaları [41, 46-53] analiz etmişler ve C&K metrik seti için, Tablo 8'de gösterilen eşik değerleri belirlemişlerdir. Daha sonra, güvenilirliği hesaplamak için Tablo 9'da gösterilen kuralları ve formülleri tanımlamışlardır. Bu çalışmada, güvenilirliğin değerlendirmek amacıyla [29, 30, 40] çalışmalarında öngörülen adımlar temel alınmıştır.

4.3.1.3. Metrik verilerinin toplanması ve kod-tabanlı değerlendirmenin yapılması (Adım 3.1.3.)
(Gather metric data and perform code-based evaluation (Step 3.1.3.))

Bakım yapılabilirlik ve güvenilirlik için kod-tabanlı değerlendirme farklı şekilde yapılmıştır. Bakım yapılabilrlik için, adım 3.1.2'deki bütün ağırlık değerleri, değerlendirme için belirlenen adımlara girdi alınmış ve sonuçta AKY'lar için Bakım Yapılabilirlik İndeksi (BYİ) elde edilmiştir. Güvenilirlik için, Tablo 8'de verilen eşik değerleri ve Tablo 9'da verilen kurallar ve formüller uygulanarak AKY'lar için Güvenilirlik İndeksi (Gİ) elde edilmiştir. Burada yüzeysel olarak bahsedilen bu adımlar, 5. Bölüm'de uygulaması ile birlikte detaylı olarak anlatılacaktır.

4.3.2. Toplum-tabanlı değerlendirme
(Community-based evaluation)

AKY'lar kaynak kodlarının yanı sıra piyasaya sürüldüğünden itibaren düzenli olarak kaydedilen tarihi verilere (historical data) ulaşma imkanı da sunar. Bu veriler Concurrent Version Control (CVS), Problem Reporting Database (BugDB) ya da Bugzilla gibi veri tabanlarında kaydedilir. Bu veri tabanlarında birçok veri kaydedildiği için toplum-tabanlı değerlendirme yapılırken ihtiyaçların nasıl belirleneceği ve ihtiyaçlar doğrultusunda bu verilerin nasıl kullanılacağı gerçekten karmaşık bir durumdur. Toplum-tabanlı değerlendirme yaparken bu karmaşık durumla başetmek için, belirlenen özniteliklerin ölçülmesinde bizlere yardım edecek bilgi ihtiyaçlarının net bir şekilde belirlenmesi gerekir. Bu aşamada bilgi ihtiyaçlarının belirlenmesinden metriklerin elde edilmesine ve ölçümlerin yapılmasına kadarki süreçte, ölçümün amacına uygun ve gerçekçi olması için, uluslararası bir standart olan ISO/IEC 15939 ölçüm süreci kullanılmıştır. Bu standart hangi ölçüm bilgisinin gerekli olduğu, ölçümlerin ve analiz sonuçlarının nasıl uygulanacağı gibi detayların tanımlanması konusunda yol göstericidir.

Tablo 8. Güvenilirliğin değerlendirilmesi için kullanılan metriklerin eşik (threshold) değerleri [30, 38]
(Threshold values of the metrics used to evaluate reliability [30, 38])

C&K Metrik	CBO	DIT	LCOM	NOC	RFC	WMC
Eşik değeri	3-9	1-6	1-3	1-3	6-36	6-30

Tablo 9. Güvenilirliğin hesaplanması için kullanılan kurallar ve formüller [37-40]
(Rules and formulas for calculating reliability [37-40])

Kural 1:
Eğer $(alt\ eşik\ değer\ \leq\ metrik\ değeri\ \leq alt\ ve$ üst *eşik değerin ortalaması)* *ise,*
Ağırlık =1
Kural 2:
Eğer $(alt\ ve$ üst *eşik değerin ortalaması* $\leq\ metrik\ değeri\ \leq$ üst *eşik değer) ise,*
Ağırlık =2
Kural 3:
Eğer $(Metrik\ değeri\ eşik\ değerlerinin\ dışında)$ *ise,*
Ağırlık =7
Kural 4:
NOC için (bu metrik diğer metriklerden ayrı değerlendirilecek),
G^E*(Max) değeri hesaplanırken* $log((üst\ eşik\ değer))^2$ *değeri dikkate alınır.*
G^E*(Min) değeri hesaplanırken* $log\ ((alt\ eşik\ değer))^2$ *değeri dikkate alınır.*
Kural 1'den kural 4'e doğru uygulandığında:
$G^E(Max) = k * (1/AD(WMC) + AD(DIT) + AD(RFC) + AD(LCOM) + AD(CBO)) + (log(Ü - Lt(NOC)))^2$ *(1)*
$G^E(Min) = k * (1/AD(WMC) + AD(DIT) + AD(RFC) + AD(LCOM) + AD(CBO)) + (log(A - Lt(NOC)))^2$ *(2)*
$K = sabit = 1;$
$G^E(Max) = 1 * (1/(1 + 1 + 1 + 1 + 1)) + (log(3))^2 = 0{,}4276$
$G^E(Min) = 1 * (1/(2 + 2 + 2 + 2 + 2)) + (log(1))^2 = 0{,}1000$
Güvenilirlik eşik değeri (G^E) $0{,}1000 < GE < 0{,}4276$ arasındadır [29, 30], [40, 41].
Eşik değeri bu değerler arasında ne kadar büyük ise ürün o kadar güvenilirdir. Eğer bu eşik değerleri dışında ise ürün hataya yatkındır.

4.3.2.1. Seçilen kalite özniteliğini ölçmek için bilgi ihtiyaçlarının belirlenmesi (Adım 3.2.1) (Identifying information needs to assess selected quality attribute (Step 3.2.1))

Tablo 10'da gösterildiği gibi bakım yapılabilirlik ve güvenilirliği ölçmek için ISO/IEC 15939 temel alınarak beş tane bilgi ihtiyacı belirlenmiştir. Bu bilgi ihtiyaçlarının ölçülebilmesi için her bir ürünün İnternet siteleri derinlemesine analiz edilmiş ve bu bilgilere nasıl ulaşılabileceği araştırılmıştır. Veri tabanları analiz edilmiş ve bilgi ihtiyaçlarını karşılamaya yönelik sonuçlar elde etmek için tablodaki metrikler belirlenmiştir.

4.3.2.2. Toplum-tabanlı değerlendirme yapmak için adımların belirlenmesi (Adım 3.2.2.) (Determine steps to perform community-based evaluation (Step 3.2.2.))

Bilgi ihtiyaçları ve metrikler belirlendikten sonra sonuçları elde etmek için adımlar belirlenmiştir. Öncelikle, her bir bilgi ihtiyacını ölçmeye yönelik belirlenen metrik değerleri kullanılarak bilgi ihtiyaçları için indeks değerleri hesaplanmıştır. Genel sonuçların sağlıklı bir şekilde elde edilebilmesi için, bilgi ihtiyaçları özelinde indeks değerlerini hesaplama şeklinin, kod-tabanlı değerlendirmedeki hesaplama şekliyle tutarlı olması gerekmektedir. Diğer bir deyişle, kod-tabanlı değerlendirmede olduğu gibi en büyük indekse sahip ürün en tercih edilebilir ürün olmalıdır. Ne var ki HÇBİ ve OHÇS için indeks değerlerinin daha büyük olması ürünler için daha iyi bir durumken HCİ, ePY ve SKHY için indeks değerlerinin daha küçük olması ürünler için daha iyi bir durumdur. Bu sebeple HCİ, ePY ve SKHY için elde edilen indeks değerlerinin, genel sonuçların doğru hesaplanabilmesi için yeniden düzenlenmesi gerekmiştir. Bunun için öncelikle, her bir ürün için hesaplanan HCİ, ePY veya SKHY değerlerinin 1 ile bölünerek tersi alınmıştır. Denklem 3'teki gibi tersi alınarak elde edilen değerlerden en büyük olanının 10 tabanında logaritması alınmış ve elde edilen değer kendisinden büyük olan en küçük tam sayıya tavan fonksiyonu [54] yardımıyla yuvarlanmıştır. Daha sonra elde edilen bu en büyük doğal sayı (K), denklem 4'te yerine konularak her bir ürün için HCİ, ePY veya SKHY bakımından yeni indeks değerleri hesaplanmıştır.

$x_1, x_2, \dots x_n$ değerlerinin sırayla $1., 2., ve\ n.$ ürün için HCİ, ePY veya SKHY bakımından elde edilen indeks değerlerini simgelediği varsayıldığında, tavan fonksiyonunu $\lceil . \rceil$: $[0,1] \subseteq \mathbb{R} \rightarrow \mathbb{Z}^+$ şeklinde tanımlarsak:

$$K = \lceil (log_{10}\, max\left(\frac{1}{x_1}, \frac{1}{x_2}, \dots, \frac{1}{x_n}\right)) \rceil \quad (3)$$

değerini elde ederiz. Li'nin HCİ, ePY veya SKHY bakımından değerlendirme yapılan $i.$ ürün için elde edilen yeni indeks değerini simgelediği varsayıldığında yeni indeks değerleri:

$$Yeni\ indeks\ değeri = L_i = \frac{1/x_i}{10^K} \quad (4)$$

eşitliği ile tanımlanabilir. Gerçekten de, her $i, j \in \{1,2,3, \dots, n\}$ için $x_i \leq x_j$ ise $L_j \leq L_i$ olduğu görülebilir. Öte yandan her $i \in \{1,2,3, \dots, n\}$ için $\frac{1/x_i}{\max\left(\frac{1}{x_1}, \frac{1}{x_2}, \dots, \frac{1}{x_n}\right))} \leq 1$ ve $0 < \frac{1/x_i}{10^{\lceil (\log_{10} \max\left(\frac{1}{x_1}, \frac{1}{x_2}, \dots, \frac{1}{x_n}\right)) \rceil}} \leq \frac{1/x_i}{10^{(\log_{10} \max\left(\frac{1}{x_1}, \frac{1}{x_2}, \dots, \frac{1}{x_n}\right))}} \leq 1$ olduğundan $0 < L_i \leq 1$ elde edilir. Bilgi ihtiyaçları için indeksler elde edildikten sonra kullanıcı ihtiyacı doğrultusunda önem derecesine göre bu bilgi ihtiyaçlarına ağırlıklar verilmiştir.

Tablo 10. Toplum-tabanlı değerlendirme için belirlenen bilgi ihtiyaçları ve metrikler
(Information needs and metrics for community-based evaluation)

Bilgi ihtiyacı (İng: Information Need)	Tanımı ve amacı	AKY veri tabanlarından kullanılan metrikler
Yıllara göre elektronik posta yoğunluğu (yıllara göre ePY)	Ürünün durağan bir e-posta trafiğinin olup olmadığını, eğer e-posta trafiği durağanlaşmışsa bunun ne kadar zaman aldığını ölçer.	Geliştiriciler arasındaki toplam e-posta sayıları, geliştiricilerin sayısı, kod satır sayısı ve her bir ürünün sürüm sayıları
Hata çözme başarı indeksi (HÇBİ)	Geliştiricilerin raporlanan hataların çözümünde ne kadar başarılı olduğunu ölçer.	Toplam hata sayısı, hatanın durumu (çözülen), toplam kod satır sayısı
Hata ciddiyet indeksi (HCİ)	Diğer aday ürünlerle karşılaştırıldığında ürünün ne derece ciddi hatalarla karşılaştığını ölçer.	Her hatanın önem derecesi ve ürünün kod satır sayısı
Ortalama hata çözme süresine göre hata çözme başarı oranı (OHÇS)	Geliştiricilerin performansını ve oluşan sorunları, hangi ürün geliştiricilerin zamanında ve disiplin içinde çözdüğünü ölçer.	Hataların açılma zamanı (son iki yıldaki), hataların kapanma zamanı (son iki yıldaki), katkı sağlayanların sayısı (number of contributors)
Ürün sürümlerine göre kümülatif hata yoğunluğu (SKHY)	Ürünlerin sürümlerine göre hataya eğilimli olup olmadığını ölçer.	Hata sayısı, toplam kod satır sayısı, her sürümün hata sayısı, her sürümün satır sayısı

4.3.2.3. Metrik verilerinin toplanması ve toplum-tabanlı değerlendirmenin yapılması (Adım 3.2.3.)
(Gather metric data and perform community-based evaluation (Step 3.2.3.))

Hesaplanan indeks değerleri ve bu ağırlık değerleri kullanılarak Toplum-tabanlı Değerlendirme İndeksi (TDİ) elde edilmiştir. Burada yüzeysel olarak bahsedilen bu adımlar, 5. Bölüm'de uygulaması ile birlikte detaylı olarak anlatılacaktır.

4.4. AKY'ların genel değerlendirmesi için adımların belirlenmesi ve seçimin yapılması
(Determine steps for general evaluation of OSS and perform selection)

Her iki boyut için değerlendirme sonuçları ayrı ayrı elde edildikten sonra kullanıcı, boyutların önem derecesini dikkate alarak bu boyutlara ağırlıklar verir (örneğin, kod-tabanlı: %50 ve toplum-tabanlı: %50). Belirlenen ağırlıklar kullanıcı ihtiyaçlarına veya ölçülmek istenen özniteliğe göre değişebilir. Örneğin, bir özniteliği ölçerken toplum-tabanlı değerlendirmede kullanılan metrikler daha belirleyici olabilir. Bu durumda toplum-tabanlı metriklerden elde edilen sonuçların ağırlığı, kod-tabanlı metriklerden elde edilen sonuçların ağırlığından fazla olacaktır. Daha sonra bu iki öznitelik (bakım yapılabilirlik ve güvenilirlik) için ayrı ayrı, iki boyutun değerlendirme sonuçlarına göre en tercih edilebilir ürün seçilir (Adım-4). Bu genel sonuç şu formül kullanılarak hesaplanır:

$$AKY\ iki\ boyutlu\ değerlendirme\ sonucu = A * B + C * D \quad (5)$$

A : Kod-tabanlı boyut için hesaplanan genel değerlendirme indeksi
B : Kullanıcı tarafından kod-tabanlı boyuta verilen ağırlık
C : Toplum-tabanlı boyut için hesaplanan genel değerlendirme indeksi
D : Kullanıcı tarafından toplum-tabanlı boyuta verilen ağırlık

5. DEĞERLENDİRME VE SONUÇLARI (EVALUATION AND RESULTS)

5.1. Kod-tabanlı değerlendirme ve sonuçları
(Code-based evaluation and results)

Tablo 5'te özellikleri gösterilen üç AKY ürününün kaynak kodları kullanılarak Tablo 7'de listelenen metrikleri sınıf seviyesinde hesaplanmıştır. Her bir ürün için bu metrik değerlerinin ortalaması Tablo 11'de verilmiştir.

5.1.1. Bakım yapılabilirliğin değerlendirilmesi ve sonuçları
(Evaluation of maintainability and results)

Bakım yapılabilirliği hesaplamak için; Tablo 7'de verilen metriklerden bakım yapılabilirliğin alt öznitelikleriyle ilişkisi olanlar, bu metriklerin Bölüm 2.3'de verilen açıklamaları ve Tablo 11'de verilen metrik değerleri kullanılmıştır. Her bir AKY ürünü her bir alt öznitelik bakımından metrik değerlerine göre, 1'den 3'e doğru (1 en tercih edilebilir ürün anlamına gelir) Tablo 12'de gösterildiği gibi sıralanmıştır. Tabloda X, Y ve Z sırayla Archiva, Maven ve Ant ürünlerini simgelemektedir. Tablonun en sağındaki

Tablo 11. Her bir AKY ürünü için metriklerin ortalama değerleri
(Average values of metrics for each OSS product)

Ürün	CBO	CC	DIT	LCOM	NNL	NOC	NOS	RFC	WMC
Archiva	3,241	1,618	1,399	2,338	0,477	1,424	21,401	12,443	17,901
Maven	2,039	2,259	1,519	3,162	0,681	1,224	28,152	7,081	20,623
Ant	3,312	2,005	1,886	2,739	0,697	1,632	21,713	22,142	26,354

Tablo 12. Bakım yapılabilirliğin alt özniteliklerine göre AKY ürünlerinin tercih edilme sıralaması
(Ordering of OSS products according to sub-attributes of maintainability)

Alt öznitelik	Sıra	WMC	DIT	RFC	NOC	NOS	NNL	CBO	LCOM	CC	Sonuç
Test edilebilirlik	1			*Y*	*Y*		*X*				Y (Maven)
	2			*X*	*X*		*Y*				X (Archiva)
	3			*Z*	*Z*		*Z*				Z (Ant)
Kararlılık	1		*X*				*X*	*Y*			X (Archiva)
	2		*Y*				*Y*	*X*			Y (Maven)
	3		*Z*				*Z*	*Z*			Z (Ant)
Değişebilirlik	1		*X*					*Y*	*X*		X (Archiva)
	2		*Y*					*X*	*Z*		Y (Maven)
	3		*Z*					*Z*	*Y*		Z (Ant)
Çözümlenebilirlik	1	*X*				*Y*				*X*	X (Archiva)
	2	*Y*				*Z*				*Z*	Y (Maven)
	3	*Z*				*X*				*Y*	Z (Ant)

kolonda, her bir alt özniteliğe göre ürünler en iyiden en kötüye doğru sıralanmıştır. Bunu yaparken tablonun her bir satırındaki ürünlerin sıra değerleri dikkate alınmıştır. Örneğin, test edilebilirliği ölçmek için üç metrik (RFC, NOC, NNL) kullanılmıştır. Bu üç metriğin değerlerine göre, iki metrik (RFC, NOC) için Maven en iyi ürün olarak gözlemlenmiş ve bir tanesi (NNL) için Archiva en iyi ürün olarak gözlemlenmiştir. Bu durumda test edilebilirlik için Maven en tercih edilebilir üründür. Bakım yapılabilirliğin her bir alt özniteliği için ürünler en iyiden en kötüye doğru, Tablo 12'deki gibi sıralanmıştır.

Ürünler her bir alt öznitelik için 1'den 3'e sıralandıktan sonra bu sıralama değerleri kullanılarak ürünlere, Tablo 13'de gösterildiği gibi ağırlıklar verilmiştir. Ağırlıklar atanırken ürünlere verilen sıra değerlerinin tersi kullanılmıştır. Örneğin Tablo 12'de 1. sırada olan ürüne ağırlık değeri olarak 3 ve 3. sırada olan ürüne ağırlık değeri olarak 1 verilmiştir. Değerlendirmenin bu kısmına kadar kullanıcı değerlendirmeye müdahale etmemiştir. Bu aşamayı takiben değerlendirici, kendisine göre daha önemli gördüğü alt özniteliklere ağırlıklar vermiştir. Bunun sebebi, bir değerlendirici için ürünün test edilebilirliği daha önemli olabilirken diğer bir değerlendirici için çözümlenebilirliğinin daha önemli olabilmesidir. Bu doğrultuda sırayla çözümlenebilirlik, test edilebilirlik, değişebilirlik ve kararlılık alt özniteliklerine, değerlendiriciye ifade ettiği önem derecesine göre 1, 2, 3 ve 4 ağırlıkları atanmıştır (verilen ağırlığın büyük olması, o özniteliğin daha önemli olduğu anlamına gelir). Atanan tüm ağırlık değerleri kullanılarak her bir ürün için Bakım Yapılabilirlik İndeksi (BYİ) hesaplanmıştır. BYİ hesaplanırken her bir alt öznitelik bakımından, ürünler için belirlenen ağırlıklar (Ağırlık Değeri(Ürün)-AD(i)) ile her bir alt özniteliğe değerlendirici tarafından verilen ağırlıklar (Ağırlık Değeri(Kullanıcı)-AD(K)) çarpılmıştır. Bu işlem her bir ürünün bütün alt öznitelikleri için yapılmış ve sonuçlar toplanmıştır. Elde edilen toplam, elde edilebilecek en büyük değere (her alt öznitelik için en büyük değer ($max(AD(i) * AD(K)) = 3 * 4 = 12$ ve toplam 4 alt öznitelik olduğundan bakım yapılabilirlik için en büyük değer $12 * 4 = 48$ olur) bölünerek BYİ elde edilmiştir. Tablo 13'te görüldüğü gibi Archiva, Maven ve Ant ürünleri için BYİ sırayla 0,583, 0,458 ve 0,208'dir.

5.1.2. Güvenilirliğin değerlendirilmesi ve sonuçları
(Evaluation of reliability and results)

Güvenilirliğin hesaplanması için doğrudan Tablo 7'de gösterilen güvenilirlikle ilişkili metrikler kullanılmıştır. Bu değerlendirme için; Tablo 11'de gösterilen metrik değerleri, Tablo 8'de gösterilen eşik değerleri ve Tablo 9'da gösterilen kurallar ve formüller temel alınmıştır. Tablo 14'te hangi ürünün hangi metrik değerine göre hangi kuralın uygulandığı ve bu kural çerçevesinde hangi ağırlık değerinin kullanıldığı gösterilmiştir.

Tablo 13. Bakım yapılabilirlik indeksi (BYİ) ve hesaplanması
(Maintainability index and its calculation)

Bakım yapılabilirlik	Archiva ağırlık değeri: $\mathbf{AD(X)}$	Maven ağırlık değeri: $\mathbf{AD(Y)}$	Ant ağırlık değeri: $\mathbf{AD(Z)}$	Kullanıcının ağırlık değeri: $\mathbf{AD(K)}$
Test edilebilirlik	2	3	1	2
Kararlılık	3	2	1	4
Değişebilirlik	3	2	1	3
Çözümlenebilirlik	3	2	1	1
	$AD(i) * AD(K)$			
Test edilebilirlik	$2 * 2 = 4$	$3 * 2 = 6$	$1 * 2 = 2$	
Kararlılık	$3 * 4 = 12$	$2 * 4 = 8$	$1 * 4 = 4$	
Değişebilirlik	$3 * 3 = 9$	$2 * 3 = 6$	$1 * 3 = 3$	
Çözümlenebilirlik	$3 * 1 = 3$	$2 * 1 = 2$	$1 * 1 = 1$	
Toplam	28	22	10	
Toplam/Mümkün olan en büyük ağırlık	28/48	22/48	10/48	
BYİ	0,583	0,458	0,208	

Tablo 14. Güvenilirlik indeksi (Gİ) hesaplanması için verilen ağırlıklar ve uygulanan kurallar
(Weights and applied rules for calculation of reliability index)

Ürün	CBO	DIT	LCOM	NOC	RFC	WMC	Gİ
Archiva	Ağırlık=1 (Kural1)	Ağırlık =1 (Kural 1)	Ağırlık =2 (Kural 2)	1,424 (Kural 4)	Ağırlık =1 (Kural 1)	Ağırlık =1 (Kural 1)	0,189
Maven	Ağırlık =7 (Kural 3)	Ağırlık =1 (Kural 1)	Ağırlık =7 (Kural 3)	1,224 (Kural 4)	Ağırlık =1 (Kural 1)	Ağırlık =2 (Kural 2)	0,062
Ant	Ağırlık =1 (Kural 1)	Ağırlık =1 (Kural 1)	Ağırlık =2 (Kural 2)	1,632 (Kural 4)	Ağırlık =2 (Kural 2)	Ağırlık =2 (Kural 2)	0,169

Tablo 9'daki denklem (1) ve denklem (2) kullanılarak her bir ürün için Güvenilirlik İndeksi (Gİ) hesaplanmıştır:

$$Archiva\ G\dot{I} = {}^{1}/_{(1+1+2+1+1)} + (\log(1{,}424))^2 = 0{,}189$$

$$Maven\ G\dot{I} = {}^{1}/_{(7+1+7+1+2)} + \log(1{,}224)^2 = 0{,}062$$

$$Ant\ G\dot{I} = {}^{1}/_{(1+1+2+2+2)} + \log(1{,}632)^2 = 0{,}169$$

Gİ değerleri incelendiğinde en güvenilir ürünün Archiva (Gİ=0,189) ve en çok hataya yatkın ürünün ise Maven (Gİ=0,062) olduğu görülmüştür.

5.2. Toplum-tabanlı Değerlendirme ve Sonuçları
(Community-based evaluation and results)

Bakım yapılabilirlik ve güvenilirliğin toplum-tabanlı değerlendirilmesi için belirlenen bilgi ihtiyaçları, açıklamaları ve bu bilgi ihtiyaçlarının hesaplanmasında kullanılacak mertikler, Tablo 10'da özetlenmişti. İzleyen alt bölümlerde her bir bilgi ihtiyacının gerekli metrikler kullanılarak nasıl ölçüldüğü anlatılacaktır.

5.2.1. Yıllara göre elektronik posta yoğunluğu (ePY)
(Electronic mail density over the years)

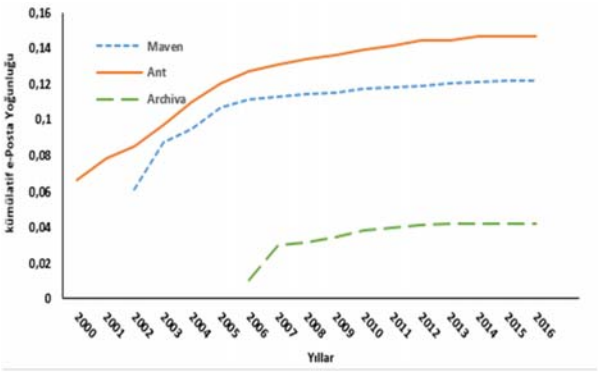
AKY'ların geliştirilmesi süresince, kod geliştiricilerin kendi aralarındaki veya kullanıcılarla kod geliştiriciler arasındaki elektronik posta (e-posta) mesajları, AKY ürünlerinin İnternet sitelerinin veri tabanlarında aylık olarak düzenli bir şekilde kaydedilir. Kod geliştirme yeteneği olan herkes bu mesajlaşma trafiğine katılabilir. E-posta mesajları ürünün geçmişi, şu anki durumu ve geleceği hakkında bilgiler (örn. ürün kodunda yapılan değişimler, karşılaşılan hatalar, önerilen değişimler, ürün hakkında teknik tartışmalar) içerir. Bu çalışmada belirlenen AKY ürünlerinin e-posta listelerine erişebilmek için, Apache'nin CVS [55] arşivi incelenmiştir.

Yıllara göre ePY hesaplamak için kod geliştiriciler arasındaki e-posta sayısı, her bir ürünün sürüm sayısı, kod satır sayısı ve kod geliştiricilerin sayısı kullanılmıştır. Ürünlerin sürüm sayıları, ürün üzerinde yapılan işlemlerin sayısı ile doğrudan ilişkilidir. Her bir yeni sürüm yeni özellikler ve önerilen yeni değişiklikler getireceği için, kod geliştiriciler arasındaki teknik tartışmaları da artıracaktır. Dolayısıyla kod geliştiriciler arasındaki e-posta sayıları da artacaktır. Kod geliştiriciler arasındaki e-posta sayıları temel alındığı için her bir üründeki kod geliştiricilerin sayısı da bu bilgi ihtiyacının hesaplanmasında önemlidir. Ayrıca her bir ürünün büyüklüğünün (kod satır sayısı) kod geliştiriciler arasındaki e-posta sayısı ile doğru orantılı olduğu varsayılabilir. Ürünün büyüklüğü arttığında ürünle ilgili problemler de artacağından e-posta sayısının da artması beklenir. Her bir ürün için ePY, toplam e-posta sayılarının toplam kod geliştirici sayılarına bölümünden elde edilen değerin (geliştirici başına e-posta sayısı), kod satır sayılarının toplam sürüm sayılarına bölümünden elde edilen değere (sürüm başına kod satır sayısı) bölünmesi ile hesaplanmıştır. Bu hesaplama sonucunda ePY için elde edilen değer ne kadar küçük ise bu, ürün için daha iyi bir durumdur. Ancak Bölüm 4.3.2'de bahsedildiği gibi, hesaplamalar yapılırken büyük indeksteki ürünlerin daha tercih edilebilir olması amaçlandığından, ePY için denklem 3 ve 4'ün uygulanması gereklidir. Denklemlerin uygulanması sonucunda değerlendirmede kullanılmak üzere elde edilen düzenlenmiş indeks değerleri, Tablo 15'te Archiva, Maven ve Ant için verildiği gibi sırayla 0,6369, 0,1388 ve 0,090 olmuştur.

Şekil 2'de yıllara göre e-posta yoğunluğu gösterilmiştir. Bu şekildeki her bir veri değeri elde edilirken her yıl için ayrı ayrı, Tablo 15'te e-posta yoğunluğunu hesaplamak için verilen formül kullanılmış ve elde edilen sonuçlar kümülatif olarak toplanmıştır. Bu şekil Tablo 15'te verilen indeks değerlerini desteklemek için elde edilmiştir.

Tablo 15. E-posta yoğunluğu indeks değerleri
(Index values of email density)

ePY	Archiva	Maven	Ant
Toplam e-posta satısı	17.395	112.776	87.438
Toplam sürüm sayısı	20	52	31
Kod geliştiricilerin sayısı	50	77	63
Kod Satır sayısı	439.346	716.983	387.863
E-posta yoğunluğu $\left(\frac{{}^{\#e-posta}/_{\#kod\ gelistici}}{{}^{KSS}/_{\#sürüm}}\right)$	0,0157	0,0720	0,110
Düzenlenmiş indeks	0,636	0,138	0,090



Şekil 2. Yıllara göre kümülatif e-posta yoğunluğu
(Cumulative email density over the years)

Şekil incelendiğinde, Archiva ürününün yıllara göre kümülatif e-posta yoğunluğunun, ilk üretiminden iki yıl sonra istikrarlı bir tutum sergilediği ve diğer iki ürüne göre hep düşük değerlerde seyrettiği görülmüştür. Bu durum, yıllara göre bu ürün ile ilgili teknik tartışmaların diğer iki ürüne göre daha az olduğunun bir göstergesidir. Ant ürününün yıllara göre e-posta yoğunluğu, diğer iki ürüne göre hep yükseklerde seyretmiştir. Yıllar içinde azalan bir seyir gösterse de ePY değerinin yüksek olması, ürünle ilgili teknik tartışmaların diğer iki ürüne göre sürekli daha fazla olduğunun bir göstergesidir. Tablo 15'ten, bu bilgi ihtiyacı

(ePY) bakımından en iyi ürünün Archiva ve en kötü ürünün Ant olduğu görülmektedir.

5.2.2. Hata çözme başarı indeksi (HÇBİ)
(Index of bug-solving success)

Yazılım hatası bilgisayar programının veya sistem akışının beklenmedik şekilde davranması veya yanlış sonuçlar üretmesidir [56]. Genellikle yazılımın kaynak kodunda veya tasarımında yapılan hatalardan meydana gelir. Bu çalışmada belirlenen üç AKY ürününün hata raporlarına ulaşmak için, problem raporu veri tabanı (BugDB) kullanılmıştır. Bu veri tabanındaki her bir hata için, hatanın açılmasından kapanmasına kadar birçok bilgi (örn. hata kimlik numarası, hatanın açıldığı ve kapandığı tarih, hatanın durumu - açıldı/çözüldü/kapandı) kaydedilir.

Tablo 16'da, her bir ürün için toplam kod satır sayısı, toplam hata sayısı, toplam çözülen hata sayısı, hata çözme başarı oranı ve hata çözme başarı indeksi verilmiştir. Hata çözme başarı oranı, her bir ürün için çözülen hata sayısının toplam hata sayısına bölünmesiyle elde edilmiştir.

HÇBİ, hata çözme başarı oranının kilo kod satır sayısına (KKSS) bölünmesiyle hesaplanmıştır. Ürünlerin kod satır sayısı ne kadar fazla ise o kadar çok hatayla karşılaşılması ve HÇBİ değerinin düşük olması beklenir. Tablo 16'ya baktığımızda en uzun kod satır sayısına sahip olan Maven'in en düşük HÇBİ değerine (0,1319) ve en az kod satır sayısına sahip Ant ürününün en yüksek HÇBİ değerine (0,2220) sahip olduğu görülmektedir.

5.2.3. Hata ciddiyet indeksi (HCİ) (Bug Severity Index)

Hata ciddiyeti (bug severity) yazılım ürünlerinde karşılaşılan hataların, ürün kalitesi üzerindeki negatif etki derecesine göre sınıflandırılmasıdır. Genellikle beş seviyeden oluşur [57] ve en önemliden en önemsize doğru sıralaması şu şekildedir: Engelleyici (blocker), kritik (critical), büyük (major), küçük (minor) ve önemsiz (trivial).

Bu bilgi ihtiyacının ölçülmesi için, problem raporu veri tabanında bulunan her bir hatanın önem derecesi incelenmiştir. Ürünlerin her bir seviyedeki hata sayıları ayrı ayrı bulunmuş ve bu hata sayıları ürünlerin kod satır sayısına bölünerek Tablo 17'de gösterildiği gibi değerler elde edilmiştir. Her bir seviyedeki hatanın ürün üzerindeki etkisi farklı olacaktır. Bu yüzden hatalara önem derecelerine göre ağırlıklar verilmiştir (engelleyici = 9, kritik = 7, büyük = 5, küçük = 3, önemsiz = 1).

Her bir ürünün HCİ değerinin hesaplanabilmesi için, Tablo 17'de gösterilen hatanın önem derecesi değerleri ile önem derecesinin ağırlıkları çarpılmış ve çarpımlar aşağıda gösterildiği gibi toplanmıştır:

$$Archiva = (0{,}00004 * 9) + (0{,}00002 * 7) + (0{,}0034 * 5) + (0{,}0005 * 3) + (0{,}00004 * 1) = 0{,}01904$$

$$Maven = (0{,}0001 * 9) + (0{,}0002 * 7) + (0{,}005 * 5) + (0{,}0011 * 3) + (0{,}0002 * 1) = 0{,}0308$$

$$Ant = (0{,}002 * 9) + (0{,}0021 * 7) + (0{,}0056 * 5) + (0{,}0049 * 3) + (0{,}0006 * 1) = 0{,}076$$

HCİ değerlerine baktığımızda Ant ürününün piyasaya sürüldüğünden itibaren diğer ürünlere göre daha ciddi hatalarla karşılaştığı görülmektedir. Archiva ürününün ise diğer ürünlere nazaran daha önemsiz hatalarla karşılaştığı gözlemlenmiştir. HCİ için elde edilen değer ne kadar küçük ise bu, ürün için daha iyi bir durumdur. Ancak Bölüm 4.3.2'de bahsedildiği gibi, hesaplamalar yapılırken büyük indeksteki ürünlerin daha tercih edilebilir olması amaçlandığından HCİ için denklem 3 ve 4'ün uygulanması gerekmektedir. Denklemlerin uygulanması sonucunda değerlendirmede kullanılmak üzere elde edilen düzenlenmiş indeks değerleri, Tablo 17'de gösterildiği gibi Archiva, Maven ve Ant için sırayla 0,52, 0,32 ve 0,13 olmuştur.

5.2.4. Ortalama hata çözme süresine göre hata çözme başarı oranı (OHÇS) (Bug-solving success rate with respect to average bug resolution time)

AKY ürünlerinin kalitesinin hesaplanmasında, ürün geliştiriciler çok önemlidir. Hangi ürünün geliştirici takımı karşılaşılan problemleri çözmede daha çabuk ve disiplinli ise o ürün daha başarılı ve güvenilir kabul edilir. Bu bilgi

Tablo 16. Hata çözme başarı indeksi (HÇBİ) (Index of bug-solving success)

Ürün	Kod satır sayısı	Toplam hata sayısı	Çözülen hata sayısı	Başarı oranı	HÇBİ (Başarı oranı/KLOC)
Archiva	439.346	1897	1663	%87,6	0,199
Maven	716.983	4914	4644	%94,5	0,131
Ant	387.863	5962	5135	%86,1	0,222

Tablo 17. Ürünlerin hata ciddiyet seviyelerindeki indeksleri ve HCİ (Index of products at bug severity levels and Bug Severity Index)

Ürün	Engelleyici	Kritik	Büyük	Küçük	Önemsiz	HCİ	Düzenlenmiş İndeks
Archiva	0,00004	0,00002	0,0034	0,005	0,00004	0,01904	0,520
Maven	0,0001	0,0002	0,005	0,0011	0,0002	0,0308	0,320
Ant	0,002	0,0021	0,0056	0,0049	0,0006	0,076	0,013

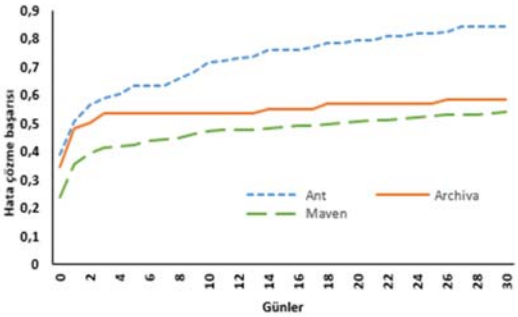
ihtiyacı, direkt olarak ürünlerdeki kod geliştiricilerin performansını ölçmektedir.

Bu çalışmada OHÇS değerini ölçmek için son iki yıldaki hatalar göz önünde bulundurulmuştur. Hataların açılış tarihi, kapanış tarihi ve bu ürünlere katkı sağlayanların sayısı (number of contributors), bu bilgi ihtiyacını ölçmek için kullanılacaktır.

Tablo 18'de son iki yılda açılan hataların her bir ürün için ortalama kaç günde çözüldüğü ve her bir üründeki katkı sağlayıcıların sayısı verilmiştir. Şekil 3'te her bir ürün için son iki yılda açılan toplam hata sayılarının açıldıktan itibaren 30 gün içerisinde çözülme oranları görülmektedir (örn. bir hata 10 Kasım 2015'te açılmış ve 11 Kasım 2015'te kapatılmış ise 1 gün içinde çözülmüştür). Son iki yılda açılan hataların çözülme oranlarına baktığımızda, Ant ürününde açılan hataların yüzde 40'ının (0,4), Archiva'da yüzde 35'inin (0,35) ve Maven'de yüzde 25'inin (0,25) açıldığı gün çözüldüğü görülmüştür.

Tablo 18. Son iki yılda açılan hataların çözülmesi ile ilgili bilgiler
(Information on solving bugs in the last two years)

Ürün	Archiva	Maven	Ant
Son iki yıldaki OHÇS	112 *gün*	106 *gün*	21 *gün*
Katkı sağlayanların sayısı	50	77	63
Son iki yılda açılan hataların açıldığı günden itibaren 30 gün içinde çözülme oranı	0,550	0,450	0,850



Şekil 3. Son iki yılda açılan hataların açıldığı günden itibaren 30 gün içerisinde çözülme oranları
(Resolution rates within 30 days of opening of the bugs in the last two years)

Genel olarak baktığımızda, hem Tablo 18'de hem de Şekil 3'te gösterildiği gibi, son iki yıl içinde açılan toplam hataların Ant ürünü için %85'i (0,85), Archiva ürünü için %55'i (0,55) ve Maven ürünü için %45'i (0,45) açıldıktan itibaren 30 gün içerisinde çözülmüştür. Dolayısıyla OHÇS bakımından indeks değerleri Archiva, Maven ve Ant için sırayla 0,55, 0,45 ve 0,85 olmuştur.

Sonuçlar incelendiğinde OHÇS bakımından Ant ürünü en başarılı ürün olmuştur. Tablo 18'de son iki yılda açılan hataların çözülmesi için gerekli ortalama süre bakımından Maven ürünü ile Archiva ürünü birbirine yakındır. Fakat Şekil 3 incelendiğinde Archiva ürününün kod geliştiricisi sayısının daha az olmasına rağmen genel anlamda hataları çözme oranında daha başarılı olduğu görülmüştür.

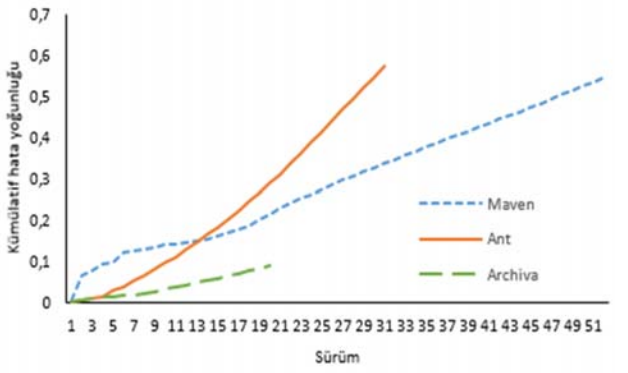
5.2.5. Ürün sürümlerine göre kümülatif hata yoğunluğu (SKHY) (Cumulative error density with respect to product versions)

Hata yoğunluğu (defect density) bir sistemde veya bileşende bulunan toplam hata sayısının o sitemin büyüklüğüne oranı olarak tanımlanır [58]. Hata yoğunluğunu azaltmak büyük projelerde zaman, maliyet ve kaliteyi dengede tutmada önemli bir yere sahiptir.

Burada belirlenen bilgi ihtiyacını (SKHY) ölçmek için, ürünlerin toplam hata sayıları ve son sürümlerinin toplam kod satır sayıları kullanılmıştır. Ürünler piyasaya sürüldüklerinden itibaren son sürümlerindeki toplam hata sayılarının, toplam kod satır sayısına bölünmesi ile elde edilen hata yoğunluğu (HY) değerleri Tablo 19'da verilmiştir. Tablodaki değerlere baktığımızda en tercih edilebilir ürün Archiva ve en hata eğilimli ürün ise Ant olarak görülmektedir. Ancak Bölüm 4.3.2'de bahsedildiği gibi tüm bilgi ihtiyaçları için elde edilen indekslerin belli bir standartta olması gerekir. SKHY için elde edilen değer ne kadar küçük ise bu, ürün için daha iyi bir durumdur. Hesaplamalar yapılırken büyük indeksteki ürünlerin daha tercih edilebilir olması amaçlandığından SKHY için denklem 3 ve 4'ün uygulanması gereklidir. Denklemlerin uygulanması sonucunda değerlendirmede kullanılmak üzere elde edilen düzenlenmiş indeks değerleri Archiva, Maven ve Ant için tablodaki gibi sırayla 0,232, 0,212 ve 0,065 olmuştur.

Tablo 19. Hata yoğunluğu (Defect density)

Ürün	Archiva	Maven	Ant
Hata yoğunluğu	0,0043	0,0047	0,0153
Düzenlenmiş indeks	0,232	0,212	0,065



Şekil 4. Sürümlere göre kümülatif hata yoğunluğu
(Cumulative defect density by versions)

Bu durumu desteklemek için her bir ürünün hata yoğunluğu Şekil 4'teki gibi sürüm bazında incelenmiştir. Şekilde her bir ürün için üretildiği yıldan itibaren her bir sürümde o ana dek bulunan hata sayısı, o sürümün toplam kod satır sayısına

bölünmüştür. Her bir sürüm için önceki sonuçlar, kümülatif olarak toplanmıştır.

Şekil incelendiğinde sürümler bazında Ant yazılım ürününün hata yoğunluğunun yüksek seviyede seyrettiği gözlemlenmiştir. Dolayısıyla hem Tablo 19'dan hem de Şekil 4'ten görüldüğü gibi, SKHY bakımından en iyi ürün Archiva ve en kötü ürün Ant olmuştur.

5.2.6. Toplum-tabanlı değerlendirme için genel sonuçlar (General results for community-based evaluation)

Toplum-tabanlı değerlendirmede, bakım yapılabilirlik ve güvenilirlik için belirlenen bilgi ihtiyaçları, seçilen metrikler doğrultusunda ölçülmüştür. Ölçüm sonuçları değerlendirilerek her bilgi ihtiyacı için indeksler elde edilmiştir. Bu indeksler metrik değerlerine göre elde edildiğinden öznellik içermemektedir.

Genel sonuca ulaşabilmek için değerlendirici, belirlediği bilgi ihtiyaçlarına kendi açısından önem derecesine göre ağırlıklar verir. Bir kullanıcı için üründe çalışan kod geliştiricilerin başarısı (OHÇS) daha önemliyken diğer bir kullanıcı için ürünün piyasaya sürüldüğünden beri ne derece önemli hatalarla karşılaştığı (HCİ) bilgisi daha önemli olabilir. Bu çalışmada ePY, HÇBİ, HCİ, OHÇS ve SKHY bilgi ihtiyaçlarına, değerlendiriciye ifade ettiği öneme göre sırayla 4, 2, 3, 1 ve 5 ağırlık değerleri verilmiştir (ağırlık ne kadar büyükse bilgi ihtiyacı o kadar önemlidir). Belirlenen ağırlıklar sonucunda her bir ürün için Tablo 20'de gösterildiği gibi Toplum-tabanlı Değerlendirme İndeksi (TDİ) hesaplanmıştır. TDİ hesaplanırken her bir bilgi ihtiyacı bakımından ürünler için ölçülen ağırlıklar (ağırlık değeri(Ürün)-AD(i)) ile her bir bilgi ihtiyacına değerlendirici tarafından verilen ağırlıklar (ağırlık değeri(Kullanıcı)-AD(K)) çarpılmıştır. Bu işlem her bir ürünün bütün bilgi ihtiyaçları için yapılmış ve sonuçlar toplanmıştır. Elde edilen toplam, elde edilebilecek en büyük değere (her bilgi ihtiyacı için en büyük değer $max(AD(i) * AD(K)) = 1 * 5 = 5$ ve toplam 5 bilgi ihtiyacı olduğundan toplum-tabanlı değerlendirme için en büyük değer $5 * 5 = 25$ olur) bölünerek TDİ elde edilmiştir. Dolayısıyla Tablo 20'de görüldüğü gibi Archiva, Maven ve Ant ürünleri için TDİ sırayla 0,247, 0,129 ve 0,080 olmuştur.

5.3. Genel Değerlendirme Sonuçları (General evaluation results)

Bakım yapılabilirlik ve güvenilirlik için değerlendirmeler kod-tabanlı ve toplum-tabanlı olmak üzere ayrı ayrı yapılmış ve sonuçlar Tablo 21'de gösterilmiştir. İki boyutlu değerlendirme sonucunda, bakım yapılabilirlik ve güvenilirlik için ayrı ayrı en iyi ürünü belirlemeden önce, değerlendirici bu boyutlara da ağırlıklar verir. Bunun sebebi, değerlendirme boyutlarının önem derecesinin kullanıcıya göre değişkenlik gösterebilmesidir. Bu çalışmada kod-tabanlı değerlendirmeye %65 ve toplum-tabanlı değerlendirmeye %35 ağırlık verilmiştir. Tablo 21'deki indeksler göz önüne alındığında, Denklem-5'te verilen genel değerlendirme sonuçlarının elde edilmesi formülüne göre, sonuçlar aşağıdaki gibidir:

Bakım yapılabilirlik için,

$$Apache\ Archiva = (0{,}583 * 0{,}65) + (0{,}248 * 0{,}35) = 0{,}464$$

$$Apache\ Maven = (0{,}458 * 0{,}65) + (0{,}131 * 0{,}35) = 0{,}342$$

Tablo 20. Toplum-tabanlı değerlendirme indeksi (TDİ) ve hesaplanması (*Index of community-based evaluation and its calculation*)

Bilgi ihtiyacı	Archiva ağırlık değeri: **AD(X)**	Maven ağırlık değeri: **AD(Y)**	Ant ağırlık değeri: **AD(Z)**	Kullanıcının ağırlık değeri: **AD(K)**
ePY	0,636	0,138	0,090	4
HÇBİ	0,199	0,131	0,222	2
HCİ	0,520	0,320	0,013	3
OHÇS	0,550	0,450	0,850	1
SKHY	0,232	0,212	0,065	5
	AD(i) * AD(K)			
ePY	$0{,}636 * 4 = 2{,}544$	$0{,}138 * 4 = 0{,}552$	$0{,}090 * 4 = 0{,}360$	
HÇBİ	$0{,}199 * 2 = 0{,}398$	$0{,}131 * 2 = 0{,}262$	$0{,}222 * 2 = 0{,}444$	
HCİ	$0{,}520 * 3 = 1{,}560$	$0{,}320 * 3 = 0{,}960$	$0{,}013 * 3 = 0{,}039$	
OHÇS	$0{,}550 * 1 = 0{,}550$	$0{,}450 * 1 = 0{,}450$	$0{,}850 * 1 = 0{,}850$	
SKHY	$0{,}232 * 5 = 1{,}160$	$0{,}212 * 5 = 1{,}060$	$0{,}065 * 5 = 0{,}325$	
Toplam	6,212	3,284	2,018	
Toplam/Mümkün olan en büyük ağırlık	6,212/25	3,284/25	2,018/25	
TDİ	0,248	0,131	0,080	

Tablo 21. Kod-tabanlı ve toplum-tabanlı değerlendirme sonuçları (Code-based and community-based evaluation results)

Değerlendirme boyutu	İndeksler	Archiva	Maven	Ant	Kullanıcının ağırlık değeri
Kod-tabanlı	BYİ	0,583	0,458	0,208	%65
	Gİ	0,189	0,062	0,169	
Toplum-tabanlı	TDİ	0,248	0,131	0,080	%35

$$Apache\ Ant = (0{,}208 * 0{,}65) + (0{,}080 * 0{,}35) = 0{,}163$$

Güvenilirlik için,

$$Apache\ Archiva = (0{,}189 * 0{,}65) + (0{,}248 * 0{,}35) = 0{,}208$$

$$Apache\ Maven = (0{,}062 * 0{,}65) + (0{,}131 * 0{,}35) = 0{,}085$$

$$Apache\ Ant = (0{,}169 * 0{,}65) + (0{,}080 * 0{,}35) = 0{,}137$$

Bu çalışmada geliştirilen metodun değerlendirme sonuçlarına göre, hem bakım yapılabilirlik (0,464) hem de güvenilirlik (0,208) için en büyük indekse sahip Archiva AKY ürünü, en tercih edilebilir ürün olmuştur.

6. UYGULAMA SONUÇLARININ DOĞRULANMASI (VERIFICATION OF EVALUATION RESULTS)

Bu bölümde, AKY'ların değerlendirilmesi için literatürde en çok referans edilen modellerden olan OSMM ve OpenBRR temel alınarak çalışmada belirlenen aday ürünler değerlendirilecek ve değerlendirme sonuçları, önerilen metodun uygulama sonuçlarıyla karşılaştırılacaktır.

6.1. OSMM uygulaması ve sonuçları
(Implementation of OSMM and results)

OSMM'nin amacı [8], AKY'ların herhangi bir organizasyon için uygun olup olmadığını belirlemektir. Model Capgemini [59] tarafından, 2003 yılında önerilmiş ve kullanıcıların geri beslemeleriyle sürekli güncellenmiştir. Modelde herhangi bir kalite modeline bağlı kalınmamış ve 3. Bölüm'de belirtildiği gibi, müşteri görüşlerini ön planda tutan iki ana indikatör tanımlanmıştır: Ürün indikatörü ve uygulama indikatörü. Ürün indikatörü, ürünün kalitesi ve olgunluğunu öngörüsel (kullanım öncesi) olarak ölçen indikatördür. Bu çalışmada önerilen metot ürüne yönelik kalite öznitelikleri bakımından öngörüsel sonuçlar verdiğinden, değerlendirme sonuçlarının doğrulanması için ürün indikatörü kullanılmıştır. Uygulama indikatörü, ürünlerin kullanımı sonrasında kullanıcı görüşleriyle ilgilendiği için, önerdiğimiz metodun sonuçlarının doğrulanmasında kullanılmamıştır.

Ürün indikatörü, Tablo 22'de gösterildiği gibi dört ana gruba ayrılmıştır ve bunların altında AKY'ların ölçümünde kullanılacak toplam 12 kriter içerir. Kullanıcı her bir AKY ürünü için 12 kritere 1, 3 ve 5 olmak üzere skorlar verir ve bu skorları ürün bazında toplayarak en olgun ürünü seçer (toplamı büyük olan en olgundur). Bu kriterlere hangi şartlara göre skorlar verildiği modelde ayrıntılı bir şekilde anlatılmıştır [8]. Modelde anlatılan kurallara göre AKY'lara verilen ağırlıklar ve bu ağırlıkların toplamı Tablo 22'de gösterilmiştir. Bazı kriterler değerlendirme yapılacak ürünün özelliğini yansıtmayabilir. Bu durumda bu kriterin skor değeri, karşılaştırılan tüm ürünler için 3 olarak verilmelidir (bu eşik değerdir ve altındaki değerler yeterli olgunluğa ulaşmamış ürünleri temsil eder). Bütün ürünlere aynı (ortalama) değer verilerek o kriterin ürünü pozitif ya da negatif olarak etkilemesi önlenmektedir. OSMM ürün indikatörüne göre Archiva AKY, en olgun ürün olarak belirlenmiştir.

Tablo 22. OSMM'ine göre AKY'ların değerlendirme sonuçları *(Evaluation results of OSS according to OSMM)*

Ürün indikatörü	Archiva	Maven	Ant
Ürün grubu			
Ürün yaşı	5	5	5
Ürün lisansı	3	3	3
İnsan hiyerarşisi	5	3	2
Satış noktası	3	3	3
Kod geliştirici topluluğu	3	5	3
Entegrasyon grubu			
Modülerlik	1	1	1
Diğer ürünlerle işbirliği	5	5	5
Standartlar	5	5	5
Kullanım grubu			
Destek	3	3	3
Dağıtım kolaylığı	5	3	3
Kabul grubu			
Kullanıcı topluluğu	3	3	3
Pazara girme	5	5	5
Toplam	**46**	**44**	**41**

6.2. OpenBRR uygulaması ve sonuçları
(Implementation of OpenBRR and results)

OpenBRR [10], Intel Corporation [60] ve Center for Open Source Investigation at Carnegie Mellon West [61] kuruluşları tarafından, kullanıcının gereksinimlerini en iyi karşılayan AKY ürününün belirlenmesi amacıyla geliştirilmiştir. Modelde AKY'ların değerlendirilmesi için 12 kategori belirlenmiştir.

OpenBRR modeli AKY'ların değerlendirmesini dört aşamada yapar. *Hızlı değerlendirme* (quick assessment) aşamasında, değerlendirilecek ürünler temel kullanım amacı dikkate alınarak hızlıca gözden geçirilir, ihtiyaç doğrultusunda olmayan ürünler elenir ve uygun adayların listesi oluşturulur. *Hedef kullanım değerlendirmesi* (target usage assessment) aşamasında kullanıcı tarafından OpenBRR modelinde bulunan 12 kategori, önem derecesine göre 1 den 12'ye doğru sıralanır (sayılar büyüdükçe önem

derecesi artar). Bu 12 kategoriden önemli görülen en fazla 7 tane olmak üzere değerlendirme kategorisi seçilir. *Veri toplama ve işleme* (data collection and processing) aşamasında, Tablo 23'te bulunan her bir kategorideki metrikler için veri toplanır. Bu veriler ışığında tabloda belirlenen kurallara göre ürünlere 1 ile 5 arasında skorlar verilir. Bu aşama en çok zaman alan aşamadır. *Veri dönüşümü* (data translation) aşamasında, her bir kategorideki skorlar toplanır ve kullanıcının ihtiyacını karşılayan en iyi AKY ürünü belirlenir. Önerilen metotla karşılaştırmaya en uygun olacak şekilde, modelde bahsedilen 12 kategori sıralanmıştır. OpenBRR modeline göre en çok 7 kategori

Tablo 23. OpenBRR modeline göre hesaplanan metrik değerleri ve değerlendirme sonuçları
(Metric values calculated according to OpenBRR model and evaluation results)

		Archiva		Maven		Ant	
Kategori	Metrik	Hesaplanan metrik değeri	Ağırlık	Hesaplanan metrik değeri	Ağırlık	Hesaplanan metrik değeri	Ağırlık
Kalite	Son 12 aydaki küçük çaplı sürüm sayısı	2	5	1	3	3	3
	Son 12 aydaki büyük çaplı sürüm sayısı	1	3	0	1	1	3
	Son 6 ayda açılan hata sayısı	16	5	83	4	52	4
	Son 6 ayda çözülen hata sayısı	6	2	47	2	13	2
	Açılan kritik hata sayısı	12	2	12	2	21	1
	Son 6 aydaki açılan kritik hataların çözülme süresi	10 gün	4	17 gün	3	15 gün	3
Destek	Son 6 aydaki e-posta sayıları	496	4	560	4	132	2
	Profesyonel desteğin kalitesi	Sadece kurulum desteği var.	3	Sadece kurulum desteği var.	3	Sadece kurulum desteği var.	3
Dokümantasyon	Çeşitli dokümantasyonların varlığı	Kurulum ve kullanım kılavuzu var.	3	Sadece metin-tabanlı kurulum dokümanı var.	2	Sadece metin-tabanlı kurulum dokümanı var.	2
	Kullanıcıların katkısı	Kullanıcıların katkı yapmasına izin verilir ve yapılan katkılar uzmanlar tarafından düzenlenir.	5	Kullanıcıların katkı yapmasına izin verilir ve yapılan katkılar uzmanlar tarafından düzenlenir.	5	Kullanıcıların katkı yapmasına izin verilir ve yapılan katkılar uzmanlar tarafından düzenlenir.	5
Topluluk	Son 6 aydaki e-posta sayıları	496	4	560	4	132	2
	Son 6 aydaki kod geliştiricilerin sayısı	18	3	25	4	21	3
Profesyonellik	Proje yönetimi	Kod geliştirici gruplar tarafından destek verilir.	3	Kod geliştirici gruplar tarafından destek verilir.	3	Kod geliştirici gruplar tarafından destek verilir.	3
	Ana geliştirici takıma üye olma zorluğu	Oldukça zordur ve öncesinde ürüne katkıda bulunmak gerekir.	3	Oldukça zordur ve öncesinde ürüne katkıda bulunmak gerekir.	3	Oldukça zordur ve öncesinde ürüne katkıda bulunmak gerekir.	3
TOPLAM			49		43		39

seçilebileceğinden ve ayrıca, bu çalışmada önerilen metotla karşılaştırılması ve ürünün kullanım öncesi değerlendirilmesi gözetilerek en uygun 5 kategori (kalite, destek, dokümantasyon, topluluk ve profesyonellik), Tablo 23'te gösterildiği gibi seçilmiştir. Bu çalışmada kullanılan 5 kategorinin (ve diğer kategorilerin) ölçülmesi için gerekli metrikler ve skor verme kriterleri modelde açık bir şekilde anlatılmıştır [10]. Bu çalışmada temel alınan AKY'lar için modelde tanımlanan metrik verileri toplanmış ve değerleri hesaplanmıştır. Hesaplanan metrik değerleri ve modelde tanımlı kurallar temel alınarak ürünlere verilen skorlar, yine Tablo 23'te gösterilmiştir. Bu skor değerleri toplanarak OpenBRR modeli bakımından en iyi ürün seçilmiştir (değerler toplamı en büyük olan kullanıcı ihtiyacına en uygun üründür). OpenBRR modeline göre de Archiva AKY, en uygun ürün olarak belirlenmiştir.

6.3. Sonuçların Karşılaştırılması (Comparison of results)

Önerilen metodu kullanarak elde ettiğimiz sonuçlarla, OSMM ve OpenBRR kullanılarak varılan sonuçlar karşılaştırılmıştır. Mevcut metotlar daha çok, genel ürün kalitesine ve olgunluğuna yönelik, tek bir ürün seçimi yapmayı öngörür. Önerilen metot ile bakım yapılabilirlik ve güvenilirlik bakımından en iyi ürünlerin belirlenmesi için sonuçlar hesaplandığından Tablo 24'te bu iki öznitelik için sonuçlar ayrı ayrı karşılaştırılmıştır. Tablo incelendiğinde, bu çalışmada önerilen metodun sonuçlarına göre Apache Archiva ürününün, hem bakım yapılabilirlik ve hem de güvenilirlik bakımından en tercih edilebilir ürün olduğu görülmektedir. OSMM ve OpenBRR sonuçları da aynı seçimi üretmiş ve önerilen metot kullanılarak elde edilen sonuçları desteklemiştir.

7. SONUÇLAR (CONCLUSIONS)

Bu çalışmada, kullanıcı ihtiyaçlarını karşılayan AKY ürününün doğru seçiminden yola çıkılmış, literatürdeki mevcut metotlar incelenerek güçlü ve eksik yönleri tespit edilmiştir. Bu tespitler temel alınarak AKY'ları hem kod-tabanlı hem de toplum-tabanlı olarak iki boyutta değerlendiren genel bir metot önerilmiştir. Önerilen metot, aday AKY olarak belirlenen üç Java tabanlı kod inşa aracının bakım yapılabilirliğini ve güvenilirliğini değerlendirmek için uygulanmıştır. Kod-tabanlı değerlendirme yapılırken oluşturulan araştırma soruları ile bu iki özniteliği ölçmeye yönelik metrikler araştırılmış ve literatürde en çok referans edilen C&K metrik seti ile NNL, CC ve NOS metrikleri kullanılmıştır. Toplum tabanlı değerlendirme yapılırken ISO/IEC 15939 ölçüm standardı temel alınmıştır. Bu standart dâhilinde, bakım yapılabilirlik ve güvenilirliği ölçmeye yönelik bilgi ihtiyaçları belirlenmiş ve AKY'ların veri tabanlarından toplanan veriler analiz edilerek karşılanmıştır. Önerilen metodun ölçüm sonuçlarının doğrulanması amacıyla, literatürde yaygın bilinen metotlardan olan OSMM ve OpenBRR kullanılmış ve önerilen metot uygulanarak elde edilen sonuçlar ile bu metotlar uygulanarak varılan sonuçlar karşılaştırılmıştır. Tüm metotlar ile yapılan değerlendirmelerde Apache Archiva, en tercih edilebilir ürün olarak belirlenmiştir. Değerlendirme ve karşılaştırma sonuçlarına göre, OSMM ve Open BRR modelleriyle elde edilen sonuçlar, hem birbirlerini hem de önerilen modelle elde edilen sonuçları doğrular niteliktedir.

Ölçümler yapılırken kod-tabanlı değerlendirmede metrik değerlerine göre ürünlere ağırlıklar verilmiştir. Bu ağırlık değerleri kullanılarak hem bakım yapılabilirlik hem de güvenilirlik için ayrı ayrı indeks değerleri elde edilmiştir. Toplum-tabanlı değerlendirme yapılırken belirlenen bilgi ihtiyaçlarını ölçmeye yönelik metrikler kullanılarak her bir bilgi ihtiyacı için indeks değerleri elde edilmiştir. İki boyut için her bir ürün bakımından ayrı ayrı indeks değerleri elde edildikten sonra, değerlendirici tarafından ölçüm boyutlarına ağırlıklar verilmiştir. Bütün bu indeks değerleri girdi olarak alınmış ve her bir ürün için tek bir indeks değeri elde edilmiştir. En büyük indekse sahip ürün en tercih edilebilir ürün olarak belirlenmiştir. Ürünlerin seçiminde kullanılmak üzere en son elde edilen indeks değerleri, ürünler için sabit indeks değerleri değildir. Bu indeks değerleri ürünlerin karşılaştırıldığı diğer ürünlere ve değerlendirme yapan kullanıcının ihtiyacına göre değişebilir. Bu sebeple elde edilen indeks değerleri, karşılaştırılan ürünler için tercih edilebilirlik sırası oluşturan, göreli indeks değerleri olarak algılanmalıdır.

Bu çalışmanın zayıf yönü olarak, önerilen metotta kullanılan ağırlıklandırma yöntemi gösterilebilir. Kod-tabanlı ve toplum-tabanlı olarak ayrı ayrı en iyi ürün belirlendikten sonra, değerlendiriciden boyutlara ağırlık vermesi beklenmiştir. Verilen ağırlıklarla, metodun kullanıcı ihtiyacına göre şekillenmesi (esnemesi) hedeflenmiştir. Bununla birlikte, kod-tabanlı ve toplum-tabanlı değerlendirmelerde farklı ürünler en tercih edilebilir ürünler olarak belirlenirse değerlendiricinin boyutlara vereceği ağırlıklar daha da büyük önem kazanacaktır. Bu zafiyeti hafifletmek amacıyla gelecek çalışmalarda, çok-kriterli karar verme yöntemlerinin önerilen metoda kullanıcı ihtiyacını gözetecek şekilde dâhil edilmeleri düşünülebilir. Çalışmada yapılan ölçümlerin, aynı ihtiyaca sahip farklı kişiler

Tablo 24. Önerilen metot ile OSMM ve OpenBRR modellerinin değerlendirme sonuçlarının karşılaştırılması
(Comparison of evaluation results of OSMM and OpenBRR models with results of proposed method)

Metotlar	Kalite Özniteliği	Archiva	Maven	Ant	En Tercih Edilebilir Ürün
Önerilen metot	Bakım yapılabilirlik	0,464	0,342	0,163	Archiva
	Güvenlirlik	0,208	0,085	0,1378	Archiva
OSMM	(Genel)	46	42	41	Archiva
OpenBRR	(Genel)	49	43	39	Archiva

tarafından yapıldığında aynı sonuçları vermesi beklenmektedir. Değerlendirmede kullanılan metrikler nesnel metriklerdir ve ölçümleri kişiye göre değişmez. Ancak metodu uygulayacak kişilerin ihtiyaçlarını tam olarak bilmesi ve AKY altyapıları hakkında bilgi sahibi olması gerekir. Ölçüm yapan kişilerin ihtiyaçlarının farklı olması durumunda ise elde edilen sonuçların farklılık göstermesi beklenebilir. Bu kısıtlara ek olarak, çalışmada önerilen metot sadece üç Java tabanlı kod inşa aracının seçimi için uygulanmıştır ve elde edilen sonuçlar bu dar kapsamda doğrulanmıştır. Daha geniş kapsamda geçerliliğin, metodu farklı ürünlerde uygulamak ve sonuçları analiz etmek suretiyle sınanması gerekir.

Bu çalışmada kod-tabanlı değerlendirme için literatürde en yaygın uygulanan [62] C&K metrik seti kullanılmıştır. Bu boyutta değerlendirmenin doğrulanması için nesneye yönelik başka metriklerle de ölçüm yapılması faydalı olacaktır. Gelecek çalışmalarda MOOD metrik setine göre ölçümler yapılması ve elde edilen sonuçların, bu çalışmanın sonuçlarıyla karşılaştırılması hedeflenmektedir. Ayrıca bu çalışmada kullanılan değerlendirme, literatürde mevcut çalışmalarda kullanılan metriklerle ve hesaplama yöntemleriyle sınırlıdır. Gelecekteki deneysel çalışmalarda, değerlendirme için metriklerin gruplanarak birlikte kullanımı ve elde edilebilecek sonuçların değerlendirmeye etkisinin sınanması hedeflenebilir.

Bakım yapılabilirlik ve güvenilirliği değerlendirmek için önerilen dört aşamalı metodun (Şekil 1) uygulamada en fazla zaman alan aşaması, 3. aşama olan metriklerin belirlenmesi, toplanması ve ölçüm yapmak için adımların belirlenmesi aşamasıdır. Metot mevcut haliyle başka AKY ürünlerinin bakım yapılabilirlik ve güvenilirliğini değerlendirilmek için kullanılacaksa yaklaşık 5 kişi-gün işgücü gerektireceği öngörülmektedir. Metodun uygulanmasını kolaylaştırmak ve değerlendiricinin yetkinliğine ilişkin gerekleri hafifletmek adına, bir araç (tool) geliştirilmesi de gelecek çalışmalarda hedeflenenler arasındadır. AKY ürünlerini değerlendirmek için bu çalışmada manuel olarak yapılan tanımlama ve hesaplama işlemlerinin bu araç ile otomatik olarak yapılmasının, değerlendirmeyi kolaylaştırması ve gereken işgücünü azaltması beklenmektedir. Bununla birlikte bu çalışmada izlenen yöntem gelecekte, bakım yapılabilirlik ve güvenilirlik dışında farklı kalite öznitellikleri için tekrarlanabilir ve kullanıcının ihtiyacına özel değerlendirmesini destekleyecek objektif unsurları ortaya çıkarmada kullanılabilir.

KAYNAKLAR (REFERENCES)

1. Open-source software, URL: https://en.wikipedia.org/wiki/Open-source_software. Erişim tarihi: Nisan 22, 2017.
2. Black Duck Software. URL : https://www.blackducksoftware.com/. Erişim tarihi Mayıs 16, 2017.
3. North Bridge. URL: http:// www .northbridge.com/. Erişim tarihi Mayıs 21, 2017.
4. K. Noyes, Senior U.S. Correspondent, PCWorld, Apr 17, 2013, URL: http://www.pcworld.com/article/2035651/open-source-is-taking-over-the-software-world-survey-says.html. Erişim tarihi Nisan 13, 2017.
5. GitHub-Cracking the Code to GitHub's Growth. URL: https://growthhackers.com/growth-studies/github. Erişim tarihi Mayıs 11, 2017.
6. Ayyıldız T.E., ve Koçyiğit A., Correlations between problem and solution domain measures of open source software, Journal of the Faculty of Engineering and Architecture of Gazi University, 32 (3), 887-900, 2017.
7. Asiala P.M., ve Matinlassi M., "Quality assurance of open source components: Integrator point of view," Bildiri Kitabı - International Computer Software and Applications Conference, 2006, vol. 2, sf. 189–192.
8. Duijnhouwer F.W., ve Capgemini C.W., Open Source Maturity Model The Usefulness of a Maturity Model Open Source in a Corporate Setting, 2003.
9. Samoladas I., Gousios G., Spinellis D., ve Stamelos I., The SQO-OSS quality model: measurement based open source software evaluation, Open Source Dev. Communities Qual., 275, 1–11, 2008.
10. Readiness Rating for Open Source. URL: http://www.openbrr.org. Erişim tarihi Mayıs 15, 2017.
11. Method for Qualification and Selection of Open Source Software (QSOS), version 1.6. URL: http://www.qsos.org./. Erişim tarihi Mayıs 15, 2017.
12. Hauge Ø., Østerlie T., Sørensen C.F., ve Gerea M., An Empirical Study on Selection of Open Source Software - Preliminary Results, Emerg. Trends Free. Source Softw. Res. Dev. 2009. FLOSS '09. ICSE Work., sf. 42–47, Mayıs 2009.
13. ISO/IEC 25010, URL: http://iso25000.com/index.php/en/iso-25000-standards/iso-25010. Erişim tarihi Mart 29, 2017.
14. ISO9126 - Software Quality Characteristics. URL: http://www.sqa.net/iso9126.html. Erişim tarihi: Mayıs 13, 2017.
15. Kurtel K., Yazılım Bakım Altkarakteristiklerinin ISO Tabanlı Modeller Kullanılarak Ölçülmesi, Doktora tezi, Trakya Üniversitesi, Edirne, 2009.
16. Chidamber S. R., ve Kemerer C. F., A Metrics Suite for Object Oriented Design, IEEE Trans. Softw. Eng., 20 (6), 476–493, 1994.
17. Ural E., Umut T., ve Feza B., Nesneye Dayalı Yazılım Metrikleri ve Yazılım Kalitesi, Yazılım Kalitesi ve Yazılım Geliştirme Araçları Sempozyumu, 2008.
18. Calp M. H., ve Arici N., Nesne Yönelimli Tasarım Metrikleri ve Kalite Özellikleriyle İlişkisi, Politek. Derg. J. Polytech. Cilt Digit. Object Identifier, 14141, 10, 9–14, 2011.
19. Jadhav A. S. ve Sonar R. M., Evaluating and selecting software packages: A review, Information and Software Technology, 51, 3, 555–563, 2009.
20. Stol K.J.ve Babar M. A., A Comparison Framework for Open Source Software Evaluation Methods, Springer, Berlin, Heidelberg, 389–394, 2010.
21. C. Ncube and J. C. Dean, The Limitations of Current Decision-Making Techniques in the Procurement of COTS Software Components, Springer, Berlin, Heidelberg, 176–187, 2002.

22. N. Yılmaz and K. Dinçer, "A Two-Dimensional Evaluation Method for Open Source Software Selection (Açık Kaynak Yazılım Seçimi için İki Boyutlu Değerlendirme Metodu)," 2016.

23. R. E. Al-Qutaish, M. I. Muhairat, B. Al-Kasasbeh, and B. M. Al-Kasasbeh, The Analytical Hierarchy Process as a Tool to Select Open Source Software, Proceedings of the 8th WSEAS International Conference on Software Engineering, parallel and distrubuted system, Cambridge, UK, 2009.

24. Stamelos I., Angelis L., Oikonomou A., ve Bleris G. L., Code quality analysis in open source software development, Inf. Syst. J., vol. 12, no. 1, sf. 43–60, Jan. 2002.

25. Wheeler D. A., How to evaluate open source software / free software programs, sf. 1–15, 2003.

26. Sung W. J., Kim J. H., ve Rhew S. Y., A Quality Model for Open Source Software Selection, Sixth Int. Conf. Adv. Lang. Process. Web Inf. Technol. (ALPIT 2007), sf. 515–519, 2007.

27. Garousi V., Investigating the success factors of open-source software projects across their lifetime, J. Softw. Eng. Stud 4., no. 1, sf. 1-15, 2009.

28. Mockus A., Fielding R. T., ve Herbsleb J., Two Case Studies of Open Source Software Development Apache and Mozilla. ACM Transactions on Software Engineering and Methodology (TOSEM) 11, no. 3 (2002): 309-346.

29. Antony J. P., ve Dev H., Estimating Reliability of Software System Using Oblect-Oriented Metrics, Int. J. Comput. Sci. Eng. Inf. Technol. Res., vol. 3, no. 2, sf. 283–294, 2013.

30. Antony P., Predicting Reliability of Software Using Thresholds of CK Metrics., Int. J. Adv. Netw., vol. 1785, no. 6, sf. 1778–1785, 2013.

31. Hevner, March, Park, and Ram, "Design Science in Information Systems Research," MIS Q., vol. 28, no. 1, p. 75, 2004.

32. R. Wieringa and H. Heerkens, "Design Science, Engineering Science and Requirements Engineering," in 2008 16th IEEE International Requirements Engineering Conference, 2008, pp. 310–313.

33. H. Göbel, S. Cronholm, and U. Seigerroth, "Towards an agile method for ITSM self-assessment : A Design Science Research Approach," 2013.

34. B. M. A Rawashdeh, "New software model for evaluating cots.pdf," J. Comput. Sci., 2006.

35. Dagpinar M., ve Jahnke J. H., Predicting maintainability with object-oriented metrics - An empirical comparison, Working Conference on Reverse Engineering, WCRE, 2003, vol. 2003–Janua, sf. 155–164.

36. Dubey S. K. ve Rana A., Assessment of maintainability metrics for object-oriented software system, ACM SIGSOFT Softw. Eng. Notes, vol. 36, no. 5, p. 1, Eylül. 2011.

37. Bakar A. D., Sultan A. B. M., Zulzalil H., ve Din J., Review on Maintainability Metrics in Open Source software, International Review on Computers and Software, 7 (3), 903-907, 2012.

38. Saraiva J., Soares S., ve Castor F., Towards a catalog of Object-Oriented Software Maintainability metrics, in International Workshop on Emerging Trends in Software Metrics, WETSoM, 2013, sf. 84–87.

39. Saraiva J. D. A. G., França M. S., Soares S. C. B., Filho F. J. C. L., ve De Souza R. M. C. R., Classifying metrics for assessing Object-Oriented Software Maintainability: A family of metrics catalogs, in Journal of Systems and Software, 2015, vol. 103, sf. 85–101.

40. Misra S. K., Sikkim E., ve Roy B., Assessment of Object Oriented Metrics for Software Reliability, vol. 4, no. 1, sf. 432–435, 2015.

41. Rosenberg L. D., Hammer T., ve Shaw J., Software metrics and reliability, Proc. 9th Int. Symp. Softw. Reliab. Eng., sf. 1–8, 1998.

42. Scitools | Build Notes. URL: https://scitools.com/build-notes/. Erişim tarihi: Mayıs 22,2017.

43. Basili V. R., Briand L. C., ve Melo W. L., A validation of object-oriented design metrics as quality indicators, IEEE Trans. Softw. Eng., vol. 22, no. 10, sf. 751–761, 1996.

44. Gyimóthy T., Ferenc R., ve Siket I., Empirical validation of object-oriented metrics on open source software for fault prediction, IEEE Trans. Softw. Eng., vol. 31, no. 10, sf. 897–910, Oct. 2005.

45. Tsai W. T., Zhang D., Chen Y., Huang H., Paul R., ve Liao N., A software reliability model for Web services, 2004, sf. 144–149.

46. Benlarbi S., El Emam K., Goel N., ve Rai S., Thresholds for Object-Oriented Measures, Proc. 11th Int. Symp. Softw. Reliab. Eng., March, p. 24, 2000.

47. Herbold S., Grabowski J., ve Waack S., Calculation and optimization of thresholds for sets of software metrics, Empir. Softw. Eng., vol. 16, no. 6, sf. 812–841, Dec. 2011.

48. Chandra D. E., ve Linda P. E., Class Break Point Determination Using CK Metrics Thresholds, Glob. J. Comput. Sci. Technol., vol. 10, no. 14, sf. 73–77, 2010.

49. Goel B. M., ve Bhatia P. K., Analysis of reusability of object-oriented systems using object-oriented metrics, ACM SIGSOFT Softw. Eng. Notes, vol. 38, no. 4, p. 1, Jul. 2013.

50. Hitz M., ve Montazeri B., Measuring Product Attributes of Object-Oriented Systems, Angew. Inform., vol. 989, sf. 124–136, 1995.

51. Tomar S. G. A., International Journal of Engineering Research and Technology IJERT, no. Vol.1-Issue 5 (Temmuz-2012). ESRSA Publ, 2012.

52. Mago J., ve Kaur P., Analysis of quality of the design of the object oriented software using fuzzy logic, in International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT2012) Proceedings published in International Journal of Computer Applications® (IJCA), 2012, sf. 21–25.

53. Musa J. D., Iannino A., and Okumoto K., Software Reliability - Measurement, prediction, application. McGraw-Hill, 1987.

54. Balcı, M. (1997) "Matematik Analiz 1", Balcı yayınları.

55. Mailing List Archive: Apache: CVS. URL: https://lists.gt.net/apache/cvs/. Erişim tarihi Mayıs 26, 2017.
56. Software bug. Wikipedia. URL: https:// en. wikipedia. org/wiki/Software_bug. Erişim tarihi Mayıs 16, 2017.
57. Software Testing Fundemental, URL: http:// softwaretestingfundamentals.com/defect-severity/. Erişim tarihi Haziran 16, 2017.
58. Software Quality – Software Testing Fundamentals. URL: http://softwaretestingfundamentals.com/ software-quality/. Erişim tarihi Mayıs 13, 2017.
59. Home | Capgemini Worldwide. URL: https://www. capgemini.com/. Erişim tarihi Temmuz 22, 2017.
60. Intel Corporation. URL: https://www.intc.com/investor-relations/default.aspx. Erişim tarihi Temmuz 22, 2017.
61. Center for Open Source Investigation-Carnegie Mellon University. URL: http://www.cmu.edu/silicon-valley /research/cosi/. Erişim tarihi Temmuz 22, 2017.
62. Jabangwe R., Börstler J., Smite D., ve Wohlin C., Empirical evidence on the link between object-oriented measures and external quality attributes: a systematic literature review, Empir. Softw. Eng., 1–54, 2014.