

## Yazılım test edilebilirliği: bir sistematik literatür haritalaması

Ebru İrge Hanoğlu<sup>1</sup>, Ayça Tarhan<sup>1</sup>, Vahid Garousi<sup>1,2</sup>

1: Yazılım Mühendisliği Araştırma Grubu (HUSE)

Bilgisayar Mühendisliği Bölümü, Hacettepe Üniversitesi, Ankara

2: Maral Yazılım Mühendisliği Danışmanlık ve Ar-Ge Corp., Calgary, Kanada

{ebru.hanoglu, atarhan, vahid.garousi}@hacettepe.edu.tr

**Özet:** Yüksek kaliteli yazılımlar hem geliştiriciler hem kullanıcılar tarafından her zaman istenen bir sonuçtur. Çok boyutlu bir kavram olan yazılım kalitesi içsel ve dışsal pek çok faktör tarafından etkilenir. Test edilebilirlik yazılım kalitesini etkileyen en önemli faktörlerden biridir. Bir yazılımın test edilebilirlik düzeyi ne kadar yüksekse, test eforu ve maliyeti o kadar düşük olacak; sonuçta ise güvenilir ve kaliteli ürünler ortaya çıkacaktır. Ne yazık ki test edilebilirlik yazılım ürünlerinin içsel bir özelliği olmadığı için doğrudan ölçülmesi mümkün değildir. Bu nedenle literatürde test edilebilirliği ölçmek adına pek çok öneriler ortaya konmuştur. Ölçümlerin çoğu kaynak kodları üzerinden gerçekleştirilmekte, ancak kodlama tamamlandıktan sonra gerek analizden kaynaklanan hataların gerek kodlama hatalarının giderilmesi daha maliyetli ve karmaşık olmaktadır. Bu makalenin amacı test edilebilirliği tahmin eden ya da ölçmeyi sağlayan modellerin varlığı üzerine bir literatür haritalaması gerçekleştirmektir.

**Anahtar sözcükler:** yazılım testi, yazılım test edilebilirliği, tahminleme modeli, sistematik literatür haritalaması

## Software testability: a systematic literature mapping

**Abstract.** High-quality software is a result desired by all users. Software quality is a multidimensional concept and is influenced by many external and internal factors. Testability is one of the most important factors affecting the quality of the software. The higher the level testability of a software, lower the testing effort and cost will be; therefore more reliable and higher quality products will ultimately emerge. Unfortunately we cannot directly measure the testability of software since it is not an intrinsic property of the software. Therefore, many proposals are set forth in the literature in order to measure the testability. Most measurement proposals are carried out based on the source code, but we note that it is more costly and complex to eliminate errors resulting from the analysis or implementation phases. The purpose of this article is to perform a literature mapping on the existing models and techniques that are targeted to measuring or predicting the testability

**Keywords:** .Keywords: Software testing, software testability, systematic mapping, systematic literature review

## 1 Giriş

McKinsey & Company ve Oxford Üniversitesi işbirliği ile 2010 yılında 5,400 bilişim projesi üzerinde gerçekleştirilen bir çalışmanın sonuçlarına göre, yazılım projelerinin %45'i planlanan bütçeyi; %7 si belirlenen geliştirme süresini aşmakta; %56'sı ise beklenen gereksinimlerin çok azını karşılayarak tamamlanabilmektedir [1]. Yazılım projelerinde görülen bu başarısızlıklar yalnızca zaman kaybına değil, çok büyük mali kayıplara da neden olmaktadır. Bu büyük kayıpların önüne geçmek için alınacak ilk tedbirlerden biri yazılımda kalitenin artırılmasına yönelik çalışmalardır. Yazılım kalitesi geliştiriciye ya da kullanıcıya göre değişebilen ve birçok boyutu olan bir kavramdır. Juran'a göre [2] kalite kullanıma uygunluk olarak tanımlanırken, Crosby [2] kaliteyi sistemin gereksinimleri karşılama düzeyi ile ölçer. Bazı durumlarda yazılımın eksiksiz ve hatasız olması kritikken, bazı durumlarda kullanım kolaylığı kalitenin ölçüsü olabilmekte ve yazılımda bulunan hataların bakım evresinde giderilmesi sorun teşkil etmemektedir.

ISO/IEC 9126-1 standardına [3] göre yazılım kalitesi; içsel kalite özellikleri, dışsal kalite özellikleri, kullanımdaki kalite olarak modellenmektedir. Bu modele göre yazılımın kalitesi işlevsellik, güvenilirlik, kullanılabilirlik, verimlilik, bakım kolaylığı gibi özellikleri üzerinden değerlendirilmektedir. Kalite açısından önemli bir kriter olan bakım kolaylığı "yazılımın değişiklik veya düzeltme isteklerine adaptasyon yeteneği" olarak tanımlanmaktadır [4] ve bu makalede üzerinde durulan test edilebilirlik özelliği, bakım kolaylığının bir alt kategorisi olarak tanımlanmıştır.

Yazılım testi, yazılımdaki hataları bulmak, riskleri tespit etmek ve mevcut uygulamayı tanımlanan en yüksek kalite seviyesine ulaştırmak için yapılan testler bütünüdür. Yazılım test faaliyetleri ile yazılımda yer alan eksiklikler ve hatalar yazılım geliştirme sürecinin erken fazlarında fark edilir ve bunların giderilmesi ile kaliteli, kullanıcıyı ve geliştiriciyi daha çok tatmin eden ürünler ortaya çıkar. Daha az maliyetle daha kaliteli ürünler ortaya koymak için yazılımın test edilebilirliğini artırmak gerekmektedir. [5]

Yazılım test edilebilirliği, bir yazılım ürününün test faaliyetlerini desteklemesinin ölçüsüdür. Bir yazılımın test edilebilirlik düzeyi ne kadar yüksekse, test eforu ve maliyeti o kadar düşük olacaktır [4].

Test edilebilirlik yazılım ürünlerinin içsel bir özelliği olmadığı için doğrudan ölçülmesi mümkün değildir. Bu nedenle yazılımın test edilebilirliğini ölçmek için metrikler, modeller ve metotlar önerilmiştir. Bu makalenin amacı test faaliyetlerini kolaylaştırmak adına, literatürde mevcut olan, yazılım test edilebilirliğini tahmin etmeye yönelik modelleri gözden geçirmektir.

Makalenin ilerleyen kısımlarında; Bölüm 2'de ilgili çalışmalar verilmiştir. Bölüm 3'de bu çalışmada kullanılan literatür haritalama yöntemi tanımlanmıştır. Bölüm 4 literatürdeki çalışmaların bulgularının belirlediğimiz araştırma soruları kapsamında analizini içermektedir. Bölüm 5'de bu çalışmamızın sonuçları ve gelecekte ileriye yönelik yapılabilecek çalışmalar anlatılmıştır.

## 2 Bağlam ve ilgili çalışmalar

Kanıtı-dayalı yaklaşım ilk olarak tıp alanında kullanılmış bir araştırma yaklaşımıdır ve daha sonra pek çok bilimsel alanda kullanılmaya başlanmıştır. Bu yaklaşımı ilk kez 2004 yılında Kitchenham ve arkadaşları “Kanıtı-dayalı yazılım mühendisliği” [6] olarak yazılım mühendisliği alanına uyarlamışlardır. Bu bağlamda kanıtlar, özelleştirilmiş bir konu ya da soru üzerine gerçekleştirilmiş en kaliteli çalışmaların sentezi ile tanımlanır. Bu sentezi gerçekleştirmenin temel metodu sistematik literatür taramaları gerçekleştirmektir.

Sistematik literatür taramaları, araştırma konusuna yönelik katkı sağlayan birincil çalışmaları değerlendiren ikincil bir çalışmadır. Daha ayrıntılı bir tanım yapmak gerekirse; sistematik literatür taramaları belirli bir soruya yanıt ya da probleme çözüm oluşturmak için, o alanda yayınlanmış tüm çalışmaların kapsamlı bir biçimde taranarak, çeşitli dâhil etme ve dışlama kriterleri kullanarak ve araştırmaların kalitesi değerlendirilerek hangi çalışmaların derlemeye alınacağı belirlenmesi, derlemeye dâhil edilen araştırmalarda yer alan bulguların sentezlenmesidir. Tablo 1, bu çalışma ile ilgili yapılmış ikincil çalışmaları göstermektedir.

**Tablo 1.** İlgili çalışmalar (bu alanda yapılan ikincil çalışmalar)

Makale Adı	Yılı	Ref.	Çalışma- nın tipi	Birincil Çal. Say.
Design for testability: a survey	1982	[7]	Normal survey	109
A survey of reliability, maintainability, supportability, and testability software tools	1991	[8]	Normal survey	233
Survey of source code metrics for evaluating testability of object oriented systems	2010	[10]	Normal survey	95
Measuring testability of object oriented design: a systematic review	2014	[11]	SLR	29
Testability and software robustness: a systematic literature review	2015	[12]	SLR	38
Testability and software performance: a systematic mapping study	2016	[13]	SM	34

## 3 Literatür haritalama metodu

Literatür taraması kapsamında incelenecek makalelere karar verilmesi adına öncül bir sistematik haritalama çalışması gerçekleştirilmiştir. Bu çalışma aşağıda belirtilen adımlar izlenerek gerçekleştirilmiştir ve akış diyagramı Şekil 1 ile gösterilmiştir:

- Haritalama sorularının tanımlanması
- Konuyla ilgili yayınlara ulaşmak için, elektronik veri tabanlarında ve diğer kaynaklarda aramaların yapılması ve incelenecek yayınların tespit edilmesi
- İçerme (include) ve dışarıda bırakma (exclude) kriterlerinin belirlenmesi
- Belirlenen kriterlere göre yayınların seçilmesi
- Yayınların içerikleri taranarak, sınıflandırmak için kullanılacak olan yayın tarihlerinin, ilişkili alanların, çalışma türlerinin ve yayın türlerinin saptanması



















