Üniversitelerdeki Yazılım Mühendisliği Eğitim Programlarında Teknik Olmayan Becerilerin Yeri: İlk Sonuçlar

Görkem Giray[1] ve Murat Paşa Uysal[2]

[1] Bağımsız Araştırmacı, İzmir, Türkiye
gorkemgiray@gmail.com
[2] Yönetim Bilişim Sistemleri Bölümü, Başkent Üniversitesi, Ankara, Türkiye
muysal@baskent.edu.tr

Özet. Teknik becerilerin yanında teknik olmayan beceriler de yazılım mühendisliğinde önemli bir yer tutmaktadır. Dolayısıyla yazılım mühendisliği mezunları iş hayatına atılırken birtakım teknik olmayan becerilere sahip olmaları başarılarını etkileyen önemli bir etken olarak karşımıza çıkmaktadır. Bu bildiri kapsamında yazılım mühendisliği eğitim programlarında teknik olmayan becerilerin yeri üzerine yapılmış olan bir sistematik haritalama çalışmasının ilk sonuçları sunulmuştur. Tanımlanan 5 araştırma sorusu, ilgili 18 çalışma incelenerek cevaplanmıştır. Eğitim programlarında en çok vurgulanan teknik olmayan becerilerin sırasıyla iletişim, takım çalışması ve sunum yapma olduğu tespit edilmiştir. Öğrencilerin bu becerileri kazanmaları için kullanılması en fazla önerilen yaklaşım proje tabanlı öğrenme olarak gözlemlenmiştir. Uygulanan yaklaşımların başarısının değerlendirilmesi için genellikle öğrencilerden anketlerle veri toplandığı görülmüştür. Teknik olmayan becerilere yazılım mühendisliği eğitim programlarında yer verilmesi konusuna araştırmacıların son yıllarda artan bir ilgisinin bulunduğu ve bu ilginin en fazla ABD'de olduğu tespit edilmiştir.

Anahtar Kelimeler: Yazılım mühendisliği eğitimi, Yazılım mühendisliği eğitim programı, Teknik olmayan beceriler, Sosyal beceriler, Sistematik haritalama çalışması.

The Role of Non-Technical Skills in Software Engineering Curriculum in Universities: Preliminary Results

Abstract. Besides technical skills, non-technical skills also play an important role in software engineering. As a result, software engineering graduates have some non-technical skills while beginning to work, which is an important factor affecting their success. In this paper, first results of a systematic mapping study on the role of non-technical skills in software engineering curriculum are presented. We responded to 5 identified research questions by analyzing 18 relevant

studies. It has been determined that the most emphasized non-technical skills in the curriculum are communication, team work, and presentation. The most recommended approach for students to gain these skills has been observed as project-based learning. It has been observed that data from the students are often collected by surveys to assess the success of the approaches implemented. It has been determined that researchers in recent years have had an increasing interest in providing non-technical skills to be included in software engineering curriculum and that this interest is mostly in the United States.

Keywords: Software engineering education, Software engineering curriculum, Non-technical skills, Social skills, Systematic mapping study.

1 Giriş

Diğer mühendislik alanlarında olduğu gibi yazılım mühendisliği alanında da teknik olmayan becerilerin önemi her geçen gün gittikçe artmaktadır. Yazılım mühendisliği süreçleri gerek mühendislerin gerekse farklı alanlardan paydaşların birlikte çalışmasını gerektirmektedir. Birçok insanın bir amaç için bir araya gelmesi bazı sosyal süreçlerin işlemesini kaçınılmaz kılmaktadır. Bu sosyal süreçler insanların teknik olmayan becerileriyle şekillendirilir; yazılım mühendisliği süreçlerini ve elde edilen sonuçları etkiler. Bu çerçevede, yazılım mühendisliği eğitiminde teknik olmayan becerilerin yeri, incelenmesi gereken önemli bir problem olarak karşımıza çıkmaktadır.

Yazılım mühendisliği sektöründe çalışan kişilerle yapılan görüşmelerde mezunların sözlü ve yazılı iletişim, müzakere, zaman yönetimi gibi becerilere sahip olmaları gerektiği belirtilmiştir [1]. Yapılan bir çalışmada mezunların bilgi ve beceri eksikliklerinin en fazla olduğu ilk iki becerinin yazılı ve sözlü iletişim becerileri olduğu gözlenmiştir [2]. Buna ek olarak, takım çalışmasının, eleştirel düşünmenin ve liderliğin de edinilmesi gereken bilgi ve beceriler arasında oldukları tespit edilmiştir. Bu eksikleri gidermek ve sektörün ihtiyaçlarını karşılamak için teknik olmayan becerilerin yazılım mühendisliği eğitim programlarında daha fazla yer alması gerektiği belirtilmektedir [3], [4], [5]. Bu doğrultuda yazılım mühendislerinin sahip olmaları gereken teknik olmayan becerilerin neler olması gerektiğini öneren çalışmalar bulunaktadır [6].

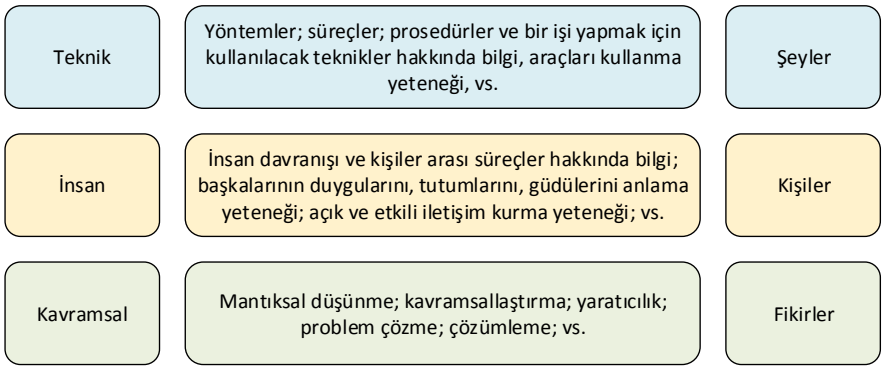
Bildirinin ikinci bölümünde çalışmanın arka planı ve ilgili çalışmalar sunulmaktadır. Üçüncü bölüm araştırma yöntemini detaylı olarak anlatmaktadır. Dördüncü bölümde araştırma sorularının cevapları yorumlarla birlikte verilmektedir. Beşinci bölümde sonuçlar ve gelecek çalışmalar sunulmaktadır.

2 Arka Plan ve İlgili Çalışmalar

Bu bölümde, teknik olmayan beceriler ve bu becerilerin sınıflandırılmaları, yazılım mühendisliği eğitim programı kılavuzlarında teknik olmayan becerilerin yeri ve ilgili çalışmalar hakkında kısaca bilgi verilecektir.

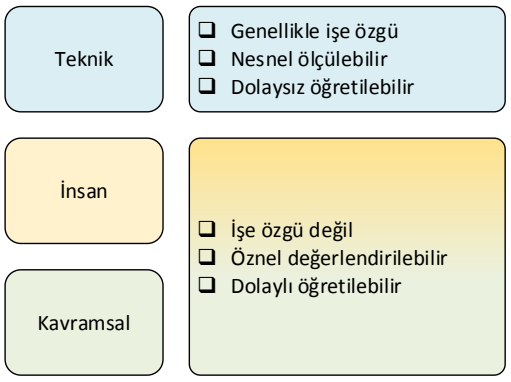
2.1 Teknik Olmayan Beceriler

Beceri, bir şeyi verimli yapabilme yeteneği olarak tanımlanmaktadır [7]. Bir insanın becerileri onun neler başarabileceğini belirler [7]. Şekil 1'de gösterildiği gibi beceriler çeşitli şekillerde sınıflandırılmaktadır. Bu sınıflandırmaya göre teknik beceriler "şeyler" ile ilgiliyken teknik olmayan beceriler (insan ve kavramsal) "kişiler" ile ve "fikirler" ile ilgilidir. Teknik olmayan beceriler İngilizce dilinde farklı şekillerde ifade edilmektedir ("soft skill", "social skill", "non-technical skill" gibi).



Şekil 1. Bilgi ve becerilerin sınıflandırılması [7].

Şekil 2'de beceri sınıflarının özellikleri gösterilmektedir. Buna göre teknik beceriler genellikle bir işe özgüdür, nesnel olarak ölçülebilir ve dolaysız öğretilebilir. Teknik olmayan beceriler ise bir işe özgü değildir, ancak öznel olarak değerlendirilebilir ve dolaylı olarak öğretilebilir. Öznel olarak değerlendirilebilme, ölçme ve değerlendirmenin zorluğuna işaret etmektedir. Dolaylı öğretilebilir olması ise doğrudan anlatmanın yeterli olmadığını ve öğrenme sürecinin öğrencinin de aktif olarak katılımını gerektirmesinin ve zamana yayılmasının gerekliliğini ortaya koymaktadır. Bu da teknik becerilerin öğretilmesinde farklı yaklaşımların ve yöntemlerin kullanılma zorunluluğunu ortaya çıkarmaktadır.



Şekil 2. Beceri sınıflarının özellikleri [7], [8].

2.2 Eğitim Programlarında Teknik Olmayan Beceriler

ACM/IEEE'nin önerdiği yazılım mühendisliği [9], bilgisayar mühendisliği [10] ve bilgisayar bilimleri [11] eğitim programlarında teknik olmayan becerilere yer verilmektedir. Yazılım mühendisliği eğitim programında mesleki uygulama başlığı, yazılım mühendisliğinin profesyonel olarak, sorumlu ve ahlaklı biçimde yürütülebilmesi için yazılım mühendislerinin sahip olmaları gereken bilgiler, beceriler ve tutumlarla ilgilidir [9]. Bu başlık altında grup dinamikleri ve psikoloji, iletişim becerileri ve profesyonellik alanları bulunmaktadır [9]. Bilgisayar mühendisliği eğitim programında tamamlayıcı beceriler başlığı altında iletişim becerileri, takım çalışması becerileri, teknik olmayan ya da kişisel beceriler, deneyim, yaşam boyu öğrenme ve iş perspektifi bulunmaktadır [10]. Tamamlayıcı becerilerin, teknik bilgilerle ve becerilerle birlikte dengeli biçimde eğitim programında yer alması gerektiği belirtilmiştir [10]. Bilgisayar bilimleri eğitim programında, öğrencilerin bilgisayar bilimlerinin toplumsal bağlamıyla karşılaşarak sosyal, ahlaki, hukuki ve profesyonel konularda beceriler geliştirmeleri gerektiği belirtilmektedir [11].

Yazılım mühendisliği alanındaki bilgi birikimini düzenleyerek belgelemeyi amaçlayan SWEBOK'un (Software Engineering Body of Knowledge) beş ana amacından biri eğitim programlarına yönelik bir temel oluşturmaktır [12]. SWEBOK'un 15 bilgi alanının birisi teknik olmayan becerilerle ilgili olan mesleki uygulama başlığına ayrılmıştır [12]. Bu başlık altındaki konular ise profesyonellik, grup dinamikleri ve psikolojisi ile iletişim becerileri olarak belirlenmiştir [12].

2.3 İlgili Çalışmalar

[13]'te bilgi teknolojileri ve bilgi sistemleri alanında çalışanların başarılı olabilmeleri için gerekli olan becerilerin bir listesi literatür taraması yapılarak oluşturulmuştur. Bir başka çalışmada mezunların becerilerinin eksikleri hakkında bulgular raporlanmıştır [2]. Bu çalışmanın amacı ise diğer tarama çalışmalarını bütünleyecek şekilde teknik olmayan becerilerin yazılım mühendisliği eğitim programlarındaki yerini incelemektir.

3 Araştırma Yöntemi

Bu çalışma kapsamında araştırma yöntemi olarak Sistematik Haritalama Çalışması (SHÇ) kullanılmıştır. SHÇ'ler, bir alandaki çalışmaları inceleyerek o alan hakkındaki bulguların genel bir değerlendirmesini sunmayı hedeflemektedir [14]. SHÇ'ler, bir alandaki çalışmaları tanımlanan araştırma soruları çerçevesinde inceleyerek sınıflandırır [14], [15]. Bu araştırmada [15]'te ve [16]'da önerilen kılavuz ve süreç kullanılmıştır. Araştırma adımları aşağıda gösterilmiştir:

1. Araştırma sorularının belirlenmesi
2. Elektronik veritabanının taranması
3. Kapsamdaki yayınların seçimi
4. Veri çıkarımı

3.1 Araştırma Soruları

Çalışmanın amacı doğrultusunda belirlenen araştırma soruları aşağıdaki gibidir:

AS1: Yazılım mühendisliği eğitim programlarında yer alan teknik olmayan beceriler nelerdir?

AS2: Yazılım mühendisliği eğitim programlarında yer alan teknik olmayan bilgi ve becerilerin edinilebilmesi için hangi öğretim yaklaşımları ve yöntemleri kullanılmaktadır?

AS3: Uygulanan yaklaşımların ve yöntemlerin doğrulaması nasıl yapılmıştır?

AS4: Çalışmalar hangi ülkelerdeki üniversitelerde yapılmıştır?

AS5: Çalışma sayısı yıllara göre nasıl bir dağılım göstermektedir?

3.2 Elektronik Veritabanının Taranması

Bu çalışma kapsamında Google Scholar veritabanı taranmıştır. Sadece bir veritabanının taranmasının nedeni bu konuda yapılmış çalışmaların bir bölümünü inceleyerek daha sonra yapılması planlanan geniş kapsamlı bir haritalama çalışmasının altyapısını hazırlamaktır. Veritabanı olarak Google Scholar'ın seçilmesinin nedeni bu veritabanının birçok diğer elektronik veritabanındaki çalışmaları içermesi ve dolayısıyla geniş bir kapsama sahip olmasıdır.

Tarama sırasında yazılım mühendisliği alanı için "software engineering", eğitim programı konusu için "curricula", "curriculum", "education", ve "program", teknik olmayan beceriler için ise "soft skill", "social skill", "non-technical skill" ve "non technical skill" sözcükleri ve sözcük öbekleri kullanılmıştır. Tarama için kullanılan arama metni aşağıdaki gibidir:

```
("software engineering" AND ("curricula" OR "curriculum" OR
"education" OR "program") AND ("soft skill" OR "social skill"
OR "non-technical skill" OR "non technical skill"))
```

3.3 Yayın Seçimi

SHÇ yöntemi doğrultusunda Tablo 1'deki eleme ölçütleri (EÖ) kullanılarak bazı çalışmalar kapsam dışında bırakılmış ve nihai kapsam belirlenmiştir.

Tablo 1. Eleme ölçütleri.

#	Açıklama
EÖ1	Bir çalışmanın listede birden fazla bulunması durumunda ilk sonuçtan sonraki çalışmalar
EÖ2	Sadece özet, çalıştay tanıtımı vb. gibi metinler içeren ve tam metin içermeyen çalışmalar
EÖ3	Yazılım mühendisliği, eğitim programı ve teknik olmayan beceriler ile bağlantılı olmayan çalışmalar
EÖ4	Kitaplar
EÖ5	Teknik rapor, tez gibi çalışmalar (bu çalışmaların bir bildiriye ya da makaleye dönüştürülerek hakemli bir konferans kitapçığında ya da bir dergide yayımlandığı varsayılmıştır)
EÖ6	İngilizce dilinde yazılmamış çalışmalar
EÖ7	Tam metnine ulaşılamayan çalışmalar

Arama sonucunda elde edilen 194 çalışmanın başlıkları, anahtar sözcükleri ve özetleri ilk yazar tarafından incelenmiştir ve eleme ölçütleri uygulanmıştır. Bazı durumlarda tam metin de gözden geçirilmiştir. Kapsama alınan ve elenen çalışmaların sayıları Tablo 2'de gösterilmektedir; 18 çalışma kapsama dahil edilirken 174 çalışma kapsam dışında bırakılmıştır.

Tablo 2. Kapsama alınan ve dışarıda bırakılan çalışma sayıları.

Kapsam içi/dışı	Çalışma sayısı
Kapsam içi	18
EÖ1	3
EÖ2	18
EÖ3	120
EÖ4	3
EÖ5	21
EÖ6	9
EÖ7	2

3.4 Veri Çıkarımı

Kapsam içinde bulunan çalışmaların tam metinleri incelenerek araştırma sorularının cevapları oluşturulmuştur. Çıkarılan veri çözümlenmek üzere Microsoft Excel programı kullanılarak düzenlenmiştir. Elde edilen sonuçlar, yazarların yorumlarıyla birlikte bir sonraki bölümde sunulmuştur.

4 Bulgular ve Tartışma

AS1: Yazılım mühendisliği eğitim programlarında incelenen teknik olmayan beceriler nelerdir?

Kapsam içindeki çalışmalarda incelenen teknik olmayan beceriler Tablo3'te gösterilmektedir. Bazı çalışmalarda [17], [18], [19] herhangi bir teknik olmayan beceri irdelenmemiştir; yapılan çalışma genel olarak teknik olmayan becerileri geliştirmeyi hedeflemektedir. Çalışmalarda en fazla iletişim becerisinin işlendiği gözlemlenmektedir. İletişim becerilerinin sözlü ve yazılı olarak ayrı ayrı ele alındığı çalışmalar da bulunmaktadır. Bunun yanında iki çalışmada [4], [20] kültürler arası iletişimin geliştirilmesine yönelik etkinlikler ele alınmıştır.

İletişimden sonra en fazla takım çalışması ve sunum yapma becerilerinin önemi vurgulanmaktadır. Özellikle son 10-15 yılda yaygınlaşan çevik yazılım geliştirme yaklaşımlarının [21] takım çalışmasına yaptığı vurgu düşünüldüğünde takım çalışmasının eğitim programlarında irdelenmesi beklenen bir sonuç olarak görülmektedir. Bunun yanında hem takım içinde hem de diğer paydaşlarla iletişim kurmanın bir yolu olan sunum yapmanın da eğitim programlarında yer almasının şaşırtıcı olmadığı düşünülmektedir.

Tablo 3. Kapsamdaki çalışmalarda incelenen teknik olmayan beceriler.

Teknik olmayan beceri	**[4]**	**[17]**	**[18]**	**[19]**	**[20]**	**[22]**	**[23]**	**[24]**	**[25]**	**[26]**	**[27]**	**[28]**	**[29]**	**[30]**	**[31]**	**[32]**	**[33]**	**[34]**
Genel		√	√	√														
İletişim (sözlü/yazılı)	√					√	√	√	√	√		√	√	√	√	√	√	
Kültürler arası iletişim	√				√													
Takım çalışması	√						√			√	√		√					√
Girişimcilik	√																	
Dinleme						√				√								
Gözlemleme						√												
Sunum yapma						√						√	√			√	√	
Müzakere						√												
Analitik düşünme						√				√								
Eleştirel düşünme						√				√								
Yenilikçilik ve yaratıcılık						√												
Özyönetim									√									
Kendi kendini organize edebilme									√									
Kendi kendini güdüleyebilme									√									

AS2: Yazılım mühendisliği eğitim programlarında incelenen teknik olmayan becerilerin edinilmesi için hangi öğretim yaklaşımları ve yöntemleri kullanılmaktadır?

Yazılım mühendisliği öğrencilerinin teknik olmayan becerilerinin üniversite öğretimi boyunca geliştirilmesi için önerilmiş olan yaklaşımlar ve yöntemler Tablo 4'te gösterilmektedir. Proje tabanlı öğrenme en fazla önerilen yaklaşımdır. Bunun yanında teknik olmayan becerilerin öğretildiği derslerin verilmesi de önerilmektedir. Genel olarak yaklaşımlara bakıldığında öğrencilerin aktif olduğu öğretim yaklaşımlarının tercih edildiği gözlenmektedir. Bunun da teknik olmayan becerilerin uygulama yapılarak geliştirildiği görüşüyle tutarlı olduğu söylenebilir.

Tablo 4. Kapsamdaki çalışmalarda önerilen yaklaşımlar/yöntemler.

Yaklaşım/Yöntem	**[4]**	**[17]**	**[18]**	**[19]**	**[20]**	**[22]**	**[23]**	**[24]**	**[25]**	**[26]**	**[27]**	**[28]**	**[29]**	**[30]**	**[31]**	**[32]**	**[33]**	**[34]**
Proje tabanlı öğrenme		√	√	√	√			√				√			√			√
Teknik derslerin içinde öğrenme	√	√																
Teknik olmayan becerilere yönelik dersler	√	√			√	√	√	√										
Kendi kendine öğrenme		√																
Problem tabanlı öğrenme			√						√					√				√
Araştırma tabanlı öğrenme			√								√		√				√	
İş ile bütünleştirilmiş öğrenme			√													√		
Kişisel görevler/ödevler						√												
Takım bazlı çalışmayla öğrenme	√					√					√							
Öğrenciler arasında tartışma ile öğrenme										√								

AS3: Uygulanan yaklaşımların ve yöntemlerin doğrulaması nasıl yapılmıştır?

Beş çalışmada [17], [18], [4], [23], [32] herhangi bir doğrulama çalışması yapılmamıştır. Diğer çalışmalarda ise yaklaşımın/yöntemin başarısı öğrencilere yapılan anketler yoluyla ölçülmüştür.

AS4: Çalışmalar hangi ülkelerdeki üniversitelerde yapılmıştır?

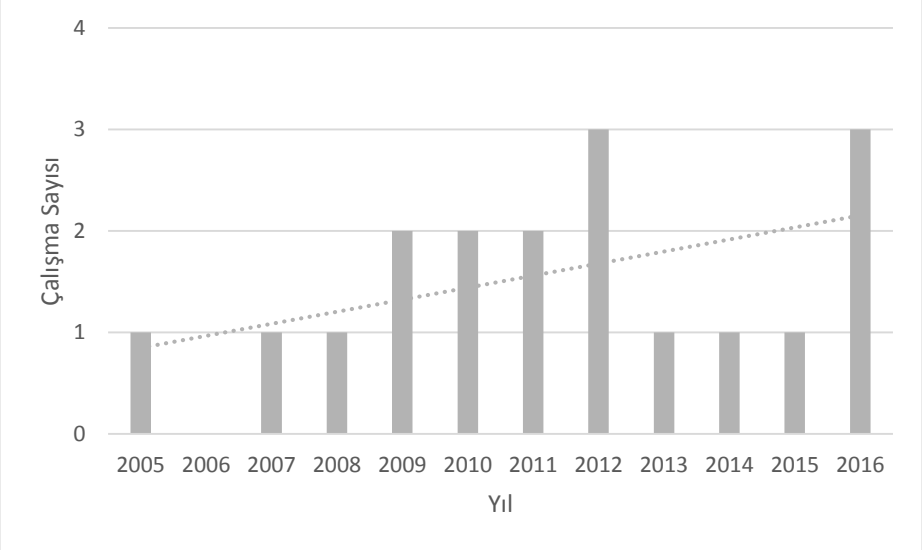
Çalışmaların yapıldığı üniversitelerin bulunduğu ülkeler Tablo 5'te listelenmektedir. Kapsamdaki çalışmaların yapıldığı üniversitelerin 7 tanesi ABD'de bulunmaktadır. ABD'yi 3 çalışma ile Avusturya ve 2 çalışmayla Çin izlemektedir. Bu çalışma kapsamında yapılan taramada Türkiye'de bu konuda yapılmış bir çalışmaya rastlanmamıştır.

Tablo 5. Çalışmaların yapıldığı üniversitelerin bulunduğu ülkeler.

Ülke	Çalışma
ABD	[19], [20], [26], [28], [29], [31], [33]
Avustralya	[17], [23], [34]
Çin	[24], [25]
Avusturya	[27]
Almanya & Rusya	[30]
İsrail	[22]
İsviçre	[4]
KKTC	[32]
Belirtilmemiş	[18]

AS5: Çalışma sayısı yıllara göre nasıl bir dağılım göstermektedir?

Çalışma sayısının yıllara göre dağılımı Şekil 3'te gösterilmektedir. Kapsam içindeki çalışmaların en eskisi 2005 yılında yapılmıştır. Son 12 yılda teknik olmayan beceriler üzerinde az da olsa artan bir ilginin olduğu gözlemlenmektedir.



Şekil 3. Çalışmaların yıllara göre dağılımı.

5 Sonuçlar ve Gelecek Çalışmalar

Son yıllarda, az da olsa artan biçimde, yazılım mühendisliği eğitim programlarında teknik olmayan becerilerin öneminin daha fazla vurgulanmaya başlandığı gözlemlenmektedir. Bu tür çalışmaların en fazla yapıldığı ülke ABD olarak tespit

edilmiştir. İletişim en fazla vurgulanan beceri olurken bunu takım çalışması ve sunum yapma becerileri izlemektedir. Öğrencilerin bu becerilerini geliştirmeleri için en fazla proje tabanlı öğrenme yaklaşımının kullanımı önerilmektedir. Teknik olmayan becerilerin ayrı bir ders olarak verilmesi de bir seçenek olarak önerilirken yaklaşımların öğrencilerin aktif olduğu yöntemlere yoğunlaştığı görülmektedir. Uygulanan yaklaşımların doğrulanması konusunda eksiklikler göz çarpmaktadır. Öğrencilere anket uygulamak doğrulama yöntemlerinden birisi olabileceği ancak tek yöntem olarak benimsenmesinin yeterli olmadığı düşünülmektedir. Bunun yanında teknik olmayan becerilerin iyi bir sınıflandırmasının olmaması ve oluşturulan listelerdeki beceriler arasındaki bağlantıların (iletişim ile sunum yapma gibi) göz ardı edilmesi bir başka araştırılacak konu olarak göze çarpmaktadır. Yazılım mühendisliği için gerekli olan teknik olmayan becerilerinin bir listesinin yapılması, bu becerilerin tanımlarının ve nasıl değerlendirilebileceğinin belgelenmesi, beceriler arasındaki bağlantıların ortaya konulması eğitim programlarının iyileştirilmesi noktasında faydalı olacaktır.

Gelecek çalışmalar kapsamında daha fazla sayıda veritabanı (ACM, IEEE Xplore, ScienceDirect gibi) taranarak daha fazla çalışmaya ulaşılması hedeflenmektedir. Böylece kapsam içindeki çalışmaların sayısı artırılarak daha genellenebilir sonuçlara varılması planlanmaktadır. Bunun yanında diğer mühendislik alanlarının eğitim programlarındaki teknik olmayan becerilerin geliştirilmesiyle ilgili yaklaşımların ve yöntemlerin incelenerek yazılım mühendisliği alanına aktarılması da çeşitli katkılar sağlayabilecektir.

Kaynakça

1. Simmons, C. B., Simmons, L. L.: Gaps in the computer science curriculum: an exploratory study of industry professionals. J. Comput. Sci. Coll. 25(5), 60-65 (2010).
2. Radermacher, A., Walia, G.: Gaps between industry expectations and the abilities of graduates. In: Proceeding of the 44th ACM technical symposium on Computer science education (SIGCSE ‘13), pp. 525-530. ACM, New York, NY, USA (2013).
3. Liem, I., Asnar, Y., Akbar, S., Mulyanto, A., Widyani, Y.: Reshaping software engineering education towards 2020 engineers. In: 2014 IEEE 27th Conference on Software Engineering Education and Training (CSEE&T), pp. 171-174. Klagenfurt (2014).
4. Pedrazzini, S.: Emphasizing Soft Skill Learning and Training as Part of an Engineering Curriculum Revision. In: Proceedings of SEFI Conference SEFI Conference, European Society for Engineering Education, Thessaloniki (2012).
5. Russell, J., Russell, B., Tastle, W. J.: Teaching Soft Skills in a Systems Development Capstone Class. Information Systems Education Journal 3(19), pp. 1-23 (2005).
6. Sedelmaier, Y., Landes, D.: Software engineering body of skills (SWEBOS). In: 2014 IEEE Global Engineering Education Conference (EDUCON), pp. 395-401. Istanbul (2014).
7. Yukl, G.: Leadership in Organizations. 6th edn. Prentice Hall, Inc. (2006).
8. Bereiter, C.: Bereiter, C., Scardamalia, M.: Education for the Knowledge Age: Design-Centered Models of Teaching and Instruction. In: Alexander, P. A., Winne, P. H. (eds.) Handbook of educational psychology, pp. 695-713. Lawrence Erlbaum Associates Publishers, (2006).
9. ACM/IEEE: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering (2014).

10. ACM/IEEE: Computer Engineering Curricula (2016).
11. ACM/IEEE: Computer Science Curricula (2013).
12. Bourque, P., Fairley, R.E. (Editörler): Guide to the Software Engineering Body of Knowledge, Version 3.0, IEEE Computer Society (2014).
13. Hagen, M., Bouchard, D.: Developing and Improving Student Non-Technical Skills in IT Education: A Literature Review and Model. Informatics 3(2), (2016).
14. Kitchenham, B., Charters, S.: Guidelines for Performing Systematic Literature Reviews in Software Engineering. EBSE 2007-001 (2007).
15. Petersen, K., Feldt, R., Mujtaba, S., Mattsson, M.: Systematic mapping studies in software engineering. In: 12th International Conference on Evaluation and Assessment in Software Engineering (EASE 2008), pp. 71–80. (2008).
16. Petersen, K., Vakkalanka, S., Kuzniarz, L.: Guidelines for conducting systematic mapping studies in software engineering: An update. Information and Software Technology 64, 1–18 (2015).
17. Makasiranondh, W., Maj, S. P., Veal, D.: Student Opinions on their Development of Non-technical Skills in IT Education. Modern Applied Science 5(2), 3-10 (2011).
18. Lowry, G., Turner, R.: Information Systems Education for the 21st Century: Aligning Curriculum Content and Delivery with the Professional Workplace. In: Carbonara, D. (ed.) Technology Literacy Applications in Learning Environments, pp. 171-202. IGI Global (2005).
19. Gary, K. A.: The Software Enterprise: Preparing Industry-Ready Software Engineers. In: Ellis, H. J. C., Demurjian, S. A., Naveda, J. F. (eds.) Software Engineering: Effective Teaching and Learning Approaches and Practices, pp. 115-135. IGI Global (2009).
20. Farley, A., Faulk, S., Lo, V., Proskurowski, A., Young, M.: Intensive international Summer Schools in Global Distributed Software Development. In: 2012 Frontiers in Education Conference Proceedings, pp. 1-6. Seattle, WA (2012).
21. Çevik Yazılım Geliştirme Manifestosu. http://agilemanifesto.org/iso/tr/manifesto.html, son erişim tarihi: 9 Haziran 2017.
22. Yadin, A.: Enhancing Information Systems Students' Soft Skill – a Case Study. I.J.Modern Education and Computer Science 10, 17-25 (2012).
23. de Salas, K., Lewis, I., Dermoudy, J., Herbert, N., Ellis, L., Springer, M., Chinthammit, W.: Designing the Modern ICT Curriculum: Opportunities and Challenges. In: 34th International Conference on Information Systems. Milan (2013).
24. Zeng, F.: Integrating Communication as a New Learning Component into Chinese Software Engineering Program. In: 2010 Annual Conference & Exposition. Louisville, Kentucky (2010).
25. Shen, G.: An Embedded System Curriculum for Undergraduate Software Engineering Program. In: Lee, R. (ed.) Software Engineering Research, Management and Applications, pp 219-232. Springer Berlin Heidelberg (2008).
26. Stuetzle, C.: Public debate format for the development of soft skill competency in computer science curricula. J. Comput. Sci. Coll. 30(6), 32-37 (2015).
27. Böhm, C., Motschnig, R.: Developing diversity awareness of software engineers: A Diversity Framework and its application in an academic and life-long learning context. In: 2016 IEEE Frontiers in Education Conference (FIE), pp. 1-8. Eire, PA (2016).
28. Bihari, T., Malkiman, I., Chaabouni, M., Bolinger, J., Ramanathan, J., Ramnath, R., Herold, M.: Enabling scalability, richer experiences and ABET-accreditable learning outcomes in computer science Capstone courses through inversion of control. In: 2011 Frontiers in Education Conference (FIE), pp. S1B-1-S1B-7. Rapid City, SD (2011).

29. Kaczmarczyk, L. C., Boutell, M. R., Last, M. Z.: Challenging the advanced first-year student's learning process through student presentations. In: Proceedings of the third international workshop on Computing education research (ICER '07), pp. 17-26. ACM, New York, NY, USA (2007).
30. Mäkiö, J., Mäkiö-Marusik, E., Yablochnikov, E.: Task-centric holistic agile approach on teaching cyber physical systems engineering. In: IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society, pp. 6608-6614. Florence (2016).
31. Brady, A.: Speaking of Software: Case Studies in Software Communication. In: Ellis, H. J. C., Demurjian, S. A., Naveda, J. F. (eds.) Software Engineering: Effective Teaching and Learning Approaches and Practices, pp. 75-97. IGI Global (2009).
32. İlkan, M., Amca, H., İşcioğlu, E.: Grooming IT Students for Industry Through Industrial Training and Graduation Project Work. Information 13(4), 1219-1242 (2010).
33. Carter, L.: Interdisciplinary computing classes: worth the effort. In: Proceedings of the 45th ACM technical symposium on Computer science education (SIGCSE '14), pp. 445-450. ACM, New York, NY, USA (2014).
34. Vivian, R., Falkner, K., Falkner, N., Tarmazdi, H.: A Method to Analyze Computer Science Students' Teamwork in Online Collaborative Learning Environments. Trans. Comput. Educ. 16(2). 2016.