İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

TAMAMLANMAMIŞ ARALIK-DEĞERLİ SEZGİSEL TERCİH İLİŞKİLERİYLE PERFORMANS DEĞERLENDİRMESİ: YAZILIM SEKTÖRÜNDE BİR UYGULAMA

YÜKSEK LİSANS TEZİ

Işılay TEZCAN

Endüstri Mühendisliği Anabilim Dalı

Mühandislik Yönetimi Programı

Tez Danışmanı: Doç. Dr. Başar ÖZTAYŞİ

HAZİRAN 2019

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

TAMAMLANMAMIŞ ARALIK-DEĞERLİ SEZGİSEL TERCİH İLİŞKİLERİYLE PERFORMANS DEĞERLENDİRMESİ: YAZILIM SEKTÖRÜNDE BİR UYGULAMA

YÜKSEK LİSANS TEZİ

Işılay TEZCAN
(507101214)

Endüstri Mühendisliği Anabilim Dalı

Mühendislik Yönetimi Programı

Tez Danışmanı: Doç. Dr. Başar ÖZTAYŞİ

HAZİRAN 2019

İTÜ, Fen Bilimleri Enstitüsü'nün **507101214** numaralı Yüksek Lisans Öğrencisi **Işılay TEZCAN**, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı "**TAMAMLANMAMIŞ ARALIK-DEĞERLİ SEZGİSEL TERCİH İLİŞKİLERİYLE PERFORMANS DEĞERLENDİRMESİ: YAZILIM SEKTÖRÜNDE BİR UYGULAMA**" başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

Tez Danışmanı :	**Doç. Dr. Başar ÖZTAYŞİ** İstanbul Teknik Üniversitesi	.....
Jüri Üyeleri :	**Prof. Dr. Cengiz KAHRAMAN** İstanbul Teknik Üniversitesi	.....
	Prof. Dr. Selçuk ÇEBİ Yıldız Teknik Üniversitesi	.....

Teslim Tarihi : 03 Mayıs 2019
Savunma Tarihi : 12 Haziran 2019

Ablama ve arkadaşlarıma,

ÖNSÖZ

Çalışmam süresince zamanını ayırıp bana destek olan, beni yönlendiren ve kendimi geliştirmeme yardımcı olan değerli danışmanım Doç. Dr. Başar Öztayşi'ye çok teşekkür ederim.

Ayrıca en zorlu zamanlarımda destek ve ilgisi ile her daim yanımda olan, çalışmayı tamamlamam için gerekli motivasyon ve yardımı esirgemeyen çok sevgili ablam Işıl Tezcan ve değerli çalışma arkadaşım Baran Demiray'a teşekkürlerimi borç bilirim.

Mayıs 2019

Işılay Tezcan
Telekomünikasyon Mühendisi

İÇİNDEKİLER

Sayfa

KISALTMALAR

AAS **:** Analitik Ağ Süreci (Analytic Network Process)

ADSBK **:** Aralık-Değerli Sezgisel Bulanık Kümeler (Interval-Valued Intuitionistic Fuzzy Set)

ADSBS **:** Aralık-Değerli Sezgisel Bulanık Sayı (Interval-Valued Intuitionistic Fuzzy Number - IVIFN)

AE **:** Yaklaşık Eşit (Approximately Equal)

AH **:** Kesinlikle Yüksek (Absolutely High)

AHP **:** Analytic Hierarchy Process

AHS **:** Analitik Hiyerarşi Süreci

ASBAB **:** Aralık-Değerli Sezgisel Bulanık Ağırlıklandırılmış Birleştirme (Interval-Valued Intuitionistic Fuzzy Weighted Aggregation)

AL **:** Kesinlikle Düşük (Absolutely Low)

ÇKKV **:** Çok Kriterli Karar Verme

DAD **:** Disciplined Agile Delivery (Disiplinli Çevik Teslimat)

EE **:** Tamamen Eşit (Exactly Equal)

FDD **:** Feature Driven Development (Özelliğe Dayalı Geliştirme)

FNA **:** Fonksiyonel Nokta Analizi (Functional Point Analysis)

H **:** Yüksek (High)

IFS **:** Intuitionistic Fuzzy Set (Sezgisel Bulanık Küme)

IIVIPR **:** Incomplete Interval-Valued Intuitionistic Preference Relation (Tamamlanmamış Aralık-Değerli Sezgisel Tercih İlişkisi)

IVIFS **:** Interval-Valued Intuitionistic Fuzzy Set (Aralık-Değerli Sezgisel Bulanık Küme)

L **:** Düşük (Low)

LeSS **:** Large Scale Scrum (Geniş Ölçekli Scrum)

MH **:** Orta Yüksek (Medium High)

ML **:** Orta Düşük (Medium Low)

PROMETHEE **:** Değerlendirmenin Zenginleştirilmesi için Tercih Sırası Düzenleme Metodu (Preference Ranking Organization METHod for Enrichment of Evaluation)

SAB **:** Sıralanmış Ağırlıklandırılmış Birleştirme (Ordered Weighted Aggregation - OWA)

SAFe **:** Scaled Agile Framework (Ölçekli Çevik Çerçeve)

TADSTİ **:** Tamamlanmamış Aralık-Değerli Sezgisel Tercih İlişkisi (Incomplete Interval-Valued Intuitionistic Preference Relation)

VH **:** Çok Yüksek (Very High)

VL **:** Çok Düşük (Very Low)

ÇİZELGE LİSTESİ

Sayfa

ŞEKİL LİSTESİ

Sayfa

TAMAMLANMAMIŞ ARALIK-DEĞERLİ SEZGİSEL TERCİH İLİŞKİLERİYLE PERFORMANS DEĞERLENDİRMESİ: YAZILIM SEKTÖRÜNDE BİR UYGULAMA

ÖZET

Bilgi teknolojileri sektöründeki hızlı gelişim ve sıkı rekabet, yetenekli çalışanların işe alımını ve şirkette kalmalarını sağlamayı oldukça önemli hale getirmiştir. Bu noktada çalışanların performans yönetimi, insan kaynağının verimli yönetimi ve hem çalışanların hem de şirketin sürekli gelişimini sağlamak için kritik bir role sahiptir.

Üretim yapan bir şirket gibi geleneksel organizasyon yapısına sahip bir şirkette, çalışanların görev ve sorumluluklarını ayırt etmek ve bu görev sorumluluklara göre performans değerlendirmesini yapmak bünyesinde yazılım geliştirme ekipleri bulunduran bir şirkete göre çok daha kolaydır. Sürekli değişen iş gereksinimlerine uyum sağlayıp değişikliklere hızlı bir şekilde cevap verebilmek için yazılım geliştirme ekiplerinin Scrum ve DevOps gibi yeni yazılım geliştirme ve proje yaşam döngülerini kullanmalarını gerekmektedir. Bu yeni yaklaşımlar, çalışanların bireysel bir rol oynayıp kahraman olmaya çalışmalarındansa takım çalışmasını daha ön plana koymuş, çalışanların teknik yeterliliklerin yanı sıra sosyal becerilerin de iyi olmasını gerektirmiştir. Çalışanların çapraz fonksiyonlu alanlarda teknik beceriye sahip olmalarına ek olarak; yaratıcı düşünceye sahip, hızlı uyum sağlayabilen, ekibi yönlendirebilen, kendi kendine organize olabilen, hevesli ve iyi iletişim kurabilen birer takım oyuncusu olmaları beklenmektedir. Bu durumda, organizasyonların da performans yönetim sistemlerini, değişen çalışma biçimlerine uygun olarak uyarlamaları gerekmektedir.

Çevik yöntemleri uygularken bireysel performansı nicel olarak ölçmek zor olduğundan birçok yazılım geliştirme şirketi verimli ve doğru bir şekilde performans değerlendirmesi yapmakta zorlanmaktadır. Çalışanlarından yeni yeteneklere sahip olmalarını beklerken geleneksel performans değerlendirme yöntemlerini kullanan organizasyonlarda, çalışanların asıl yetenekleri performans sonuçlarına yansımamaktadır. Bu nedenle, yazılım geliştirme ekiplerinde bireysel performans değerlendirmesi yaparken kullanılan performans değerlendirme kriterlerinin çevik yazılım geliştirme prensipleriyle çelişmemesi ve kriter ağırlıklarının kriterlerin proje yaşam döngüsündeki önemine göre hesaplanması oldukça önemlidir.

Performans değerlendirmesi, karar vericilerin birden fazla kriteri göz önünde bulundurarak tercih yaptıkları bir karar verme sürecidir. Çevik metotlarla yazılım geliştiren ekiplerde, hangi perfromans kriterinin bireysel performansın ve takım performansının artmasına daha fazla katkı yaptığını ölçmek, kriterlerin çoğunun net sayılarla ifade edilebilen bir etkisi ve çıktısı olmadığından oldukça zordur. Ayrıca, yazılım geliştirme ekiplerinde performans ölçümünün zor olmasının yanı sıra, karar verme sürecinin kendisi, insan düşüncesinin belirsizliğini ve tereddütlerini içerdiği için karmaşık bir problemdir. Bu karmaşıklığın üstesinden gelmek için Analitik Hiyerarşi Süreci (AHS) ve Zadeh tarafından geliştirilen bulanık küme teorisi

literatürde yaygın bir şekilde kullanılmıştır. Ancak Atanassov tarafından geliştirilen ve üyelik derecesi, üye olmama derecesi ve tereddüt derecesi ile ifade edilen sezgisel bulanık küme kavramı bulanıkla mücadele etmek için daha esnek bir çözüm sağlar. Daha sonra Atanassov ve Gargov tarafından geliştirilen aralık-değerli sezgisel bulanık küme kavramıyla, üyelik ve üye olmama derecelerini kesin ve gerçek değerlerle ifade etme zorluğu aşılmıştır. Xu'nun aralık-değerli sezgisel bulanık kümeleri, çok kriterli karar verme alanına uygulaması ve Tamamlanmamış Aralık-Değerli Sezgisel Tercih İlişkilerini (TADSTİ) tanımlamasıyla birlikte, karar vericinin üstündeki zaman baskısı veya karar vericinin yeterli bilgi ve deneyime sahip olmaması gibi problemlerinin önüne geçilmesi hedeflenmiştir. TADSTİ'ler sayesinde sadece bir kriterin diğer kriterlere göre öneminin karşılaştırılmasıyla kriter ağırlıklarını hesaplamak mümkün olmuştur. Bu çalışmada da uygulama olarak, uluslararası bir firmanın Arge departmanında faaliyet gösteren bir yazılım geliştirme test ekibinin performans ölçümü ele alınmıştır. 4 ana kriter ve 12 alt kriterin ağırlığı TADSTİ'lerle hesaplandıktan sonra, Öztayşi ve diğ. tarafından PROMETHEE (Değerlendirmenin Zenginleştirilmesi için Tercih Sırası Düzenleme Metodu)'den türetilen performans gösterge fonksiyonlarından tip-III ve Tip V, kriterlere uygun performans gösterge fonksiyonu olarak seçilmiş ve her bir kriter için elde edilen sonuçların ağırlıklı toplamıyla total performans skoru hesaplanmıştır.

TADSTİ'lerle yapılan hesaplamalara göre söz konusu ekip için çıktı kalitesi ile teknik yeterlilik ve sorun çözme kriterleri yüksek öneme sahipken; dürüstlük ve görevlerin tamamlanma süresi, ekip çalışmasına katkı, kendini adama, genel davranış, kendini geliştirme ve zorlu görevlerin başarımı orta derecede önemli kriterler; yönlendiricilik ve sorumluluk alma, çoklu görevleri başarabilme, bilgi paylaşımı ve stres yönetimi nispeten daha düşük öneme sahip kriterler olarak görülmüştür.

Bu çalışmadan önce söz konusu ekibin liderleri performans ölçümünü yaparken her bir kritere eşit ağırlık vererek hesaplama yaptığından, toplam performans skoru çalışanların gerçek performansını yansıtmamaktaydı. Bulanık küme teorisinden türetilen Tamamlanmamış Aralık-Değerli Sezgisel Bulanık Küme teorisinin uygulanışıyla birlikte hesaplanan performans skoru takım liderlerine daha doğru bir veri sağlamıştır.

Ayrıca, ağırlıkları hesaplanan performans kriterleri sadece performans skorunun doğru hesaplanmasında değil, çalışanlarla birlikte bir sonraki dönem için hedefler belirlenirken hangi noktalara daha fazla önem verilmesi gerektiği konusunda da fayda sağlamaktadır.

Uygulama sırasında kriterlerin ağırlıklarının belirlenmesi için yapılan hesaplamaları daha hızlı ve doğru bir şekilde yapabilmek için Java programlama diliyle bir yazılım geliştirilmiştir. Uzmanlar, tek bir kriterin diğer kriterlere göre önemini "Çok Yüksek" ve "Çok Düşük" aralığında değişen 9 dilsel terimle ifade ederek bir kıyaslama yaparak bir excel dosyası oluştururlar. Geliştirilen yazılım, bu excel dosyasını girdi olarak alıp bulanık ağırlıkları, netleştirilmiş ağırlıkları ve normalize edilmiş ağırlıkları çıktı olarak ekrana basar. Yazılım, uzman sayısına göre değişen SAB vektörünü ve kriter sayısına göre değişen diğer tüm hesaplamaları girdi olarak verilen Excel dosyası sayesinde otomatik olarak yapacaktır. Ayrıca bu yazılım sayesinde, ekibin proje gereksinimlerine göre çalışanlardan beklentisinin değiştiği durumlarda, yeni performans kriterlerine göre ağırlıkların hesaplanması için sadece excel dosyasındaki kriterleri ve ikili karşılaştırmalarını güncellemeleri yeterli olacaktır.

İleride yazılıma bir önyüz eklenerek, uzmanların karşılaştırmaları Excel yerine arayüzden yapıp, sonuçları da yine aynı arayüzde görmeleri sağlanabilir. Ayrıca yazılım, her bir kriter için uygun olan performans gösterge fonksiyonunu seçme olanağını uzmanlara sağlayarak, çalışanların toplam performans skorunu da hesaplayacak şekilde geliştirilebilir.

Gelecek çalışmalarda, performans değerlendirme problemi, tip-2 bulanık kümeler ve tereddütlü bulanık kümeler gibi yeni bulanık küme uzantılarıyla da ele alınabilir.

PERFORMANCE ASSESSMENT WITH INTERVAL-VALUED INTUITIONISTIC PREFERENCE RELATIONS: AN APPLICATION ON SOFTWARE DEVELOPMENT SECTOR

SUMMARY

Fast development and strong competition in the information technologies sector makes the employment and keep of the talented employees very important. To effectively manage the human resources, to ensure the continuous development of them and as well as the company and to improve the corporate performance, performance assesment of the employees plays a critical role.

In traditional organizations like a production company, distinguishing the roles and responsibilities and assessing the performance of the employess is much easier than an organization where software development teams have been located in. Keeping up with continuously changing business environments and responding quickly, requires software teams to use modern software development approaches and project life cycles like Scrum and DevOps (Development and Operations). With these new approaches, teamwork is more important than being a hero and social competencies are also required as much as technical skills. Employees are not just expected to have technical knowledge in cross functional areas but also they have to be a good team player, communicator, creative thinker, mentor and fast adopting, self organized and dedicated worker. At this point, organizations should adopt their performance assessment methods as changing their working styles.

Most of the software development companies are struggling to assess the performance efficiently and correctly since quantitatively measuring the individual performance is very challenging while applying agile methods. As a result, in organizations using traditional performance measuring methods but requring new talents, assesments are not generally reflecting the actual competencies of the employees. Thus, it is important that criteria to be used during individual performance evaluation of software team members, should not be conflicted with agile software development principles and their weights should be decided according to their importance in project lifecycle.

Performance assessment is a decision making process for which decision makers have to make a preference considering multiple criteria. In agile teams, this criteria should be set based on the agile requirements. By nature of these requirements, it is hard to decide which of the performance criteria would make more contibution to individual and team performance since most of them has not direct effects and outputs which can be expressed in a crisp way. Also, regardless of the difficulties of performance management in software teams, decision making itself is a complex process since it involves hesitations and ambiguity of human thoughts. . In order to overcome this complexity, Analytic Hierarchy Process (AHP) and fuzzy sets theory developed by Zadeh are widely used together in the literature. However, the concept of intuitionistic fuzzy set (IFS) which is introduced by Atanassov and defined by a membership degree, nonmembership degree, and a hesitancy degree seems more flexible and

practical to handle uncertainty and fuzziness than the fuzzy set developed by Zadeh. Atanassov and Gargov further studied the IFS, and defined the concept of an interval-valued intuitionistic fuzzy set (IVIFS) to overcome with the difficulty of expressing the membership degree and non-membership degree in the preference relation by exact real values. As each decision making process involves decision maker's preference of an alternative over another, a preference relation can be formed. Since, expressing the preference degree with intuitionistic fuzzy values are more convenient than expressing them in real numerical values, Szmidt and Kacprzyk had defined intuitionistic fuzzy preference relations. Then, Xu and Che applied the IVIFS theory to the multiple attribute decision making field and defined incomplete interval valued intuitionistic preference relations by considering the time pressure, lack of knowledge or experience of the decision makers. Thus with IIVIPR (Incomplete Interval-Valued Intuitionistic Preference Relation), it is possible to calculate weights of performance criteria just by having comparison result of one alternative to each other alternative in linguistic scale. And in this study, an application of IVIFS theory has been done for a software test team in R&D department of an international telecommunication company.

In this study, first, some definitions about performance management and recent software development approaches like Scrum and DevOps are given. Then, literature review on performance measurement of software teams and performance measument with fuzzy methods have been mentioned. In forth section of the study; fuzzy logic, fuzzy sets, some new extensions of fuzzy sets, fuzzy AHP, preference relations, performance indicator functions and application steps are given one by one. In the fifth section where the application details exist, first, 4 main criteria which are technical capabilities, social capabilities, task success and contribution to team and their 12 sub performance assessment criteria used to measure performance of a software test team are expressed and then each step of performance measurement method are explained. With this method, after calculating weights of the criteria using fuzzy incomplete interval valued intuitionistic preference relations, type III and type V performance indicator functions defined by Öztayşi et al. and derived from PROMETHEE (Preference Ranking Organization METHod for Enrichment of Evaluation) has been used to calculate performance score for each criteria and then total score has been calculated by summing weighted scores.

As start of the method application, pairwise importance comparison of the first criteria to the other criterias has been asked to 3 experts (2 test team leaders and 1 project leader) in a linguistic scale having 9 intervals ranging from "absolutely low" to "absolutely high". Then in step 2, by using incomplete interval values intuitionistic preference relations, the preference relations of other alternatives to each other have been calculated for each expert and these preference values have been aggregated in step 3. After calculating the scores of the aggregated values, in step 4, preferences are ordered and using OWA vector, fuzzy weight of each criteria has been calculated in step 5 and 6. In step 7, defuzzified and normalized main criteria weights have been calculated. As result, it is seen that between main performance criteria categories, technical capabilities has most importance and contribution to the team has least importance. In step 8, steps between 1 to 6 have been repeated for sub criterias and in step 9, sub criteria weights have been defuzzified and proportioned according to their respective normalized main criteria values. As result, it has been seen that, in this software test team where Scrum and DevOps approaches have been applied, output quality and technical competency & problem solving criteria have high importance; integrity & delivery on time, contribution to teamwork, dedication, general behaviour,

self improvement and challenging task success have medium importance and, guiding & taking initiative, multitasking, knowledge sharing and stress handling have low importance according to expert evaluations. In step 10, 2 test team leaders have evaluated each employee considering their success over each sub criteria; they gave a score using a scale ranging from 1 to 5 as 1 represents highly unsuccessful and 5 represents highly successful. Only exception to this scale is K33- challenging task success sub criteria as all employees do not have challenging task assignment. For this case, employees gets 0 performance score. In step 11, using performance indicator functions, K-33- challenging task success sub criteria related calculation has been done according to Type-V and for others type-III has been used. In last step, total performance scores of the employees has been calculated considering sub criteria weights and scores calculated and then, employees have been ordered according to their performance scores. The application of this method can make the evaluation results more scientific, accurate, and objective.

Before this study, test team leaders of this team, have been using same weights for all criteria which was causing unreliability in calculated performance score. With this Incomplete Interval-Valued Intuitionistic Fuzzy Set theory extented from Fuzzy Sets Theory, calculated weights of the criteria made calculation of total performance score more realistic. And with this total performance score, it is possible to rank employees of this team.

As criteria weights calculated with IIVIFS method helps to correct evaluation of observed performance, they also support the goal setting process for the next term by highlighting, improving which points will be more contributing to increase of individual and thus team performance.

It is important to state that during application of this method, it has been assumed that experts who are giving comparison of one criteria according to others, are making correct evaluation. Additionnaly, since an incomplete preference relation method has been applied and unknown relations have been calculated, inconsistency ratio between pairwise comparison of sub criteria will be zero. This is why it is not found necessary to conduct a consistency analysis.

In addition, during the application of IIVIFS theory to performance measurement of a software team, a software has been developed in Java programming language to make all computations during the weight calculation process which are included in steps from 2 to 10. This software takes an excel file as an input in which experts evaluations about pairwise comparison of one alternative to other alternatives in linguistic terms exist. This software will calculate OWA vector according to expert count and all other equations according to criteria count automatically taking this excel input file as reference. As output; fuzzy, defuzzified and normalized weights of the criteria have been printed in application console screen. If somehow, the team decides to change their software approach in a way that necessites the change of performance evaluation criteria, weight calculations of new criteria will be as easy as updating this excel file. In future, an interface to take expert evaluations and to show the result can be further developed. Also, calculation of total performance score of employees can be included in the software by letting experts to select correct performance indicator function for each alternative and then calculating the score of each criteria.

Also, in future studies, performance evaluation problem can be analyzed with other fuzzy set extensions like Type-II fuzzy sets and hesitant fuzzy sets and an application of these methods can be further tried.

1. GİRİŞ

Bilgi ve iletişim sektöründeki hızlı gelişim ve rekabet, bilginin üretimi ve kullanımı için gerekli nitelikli işgücü ihtiyacını daha da önemli hale getirmiş ve şirketler arasında nitelikli çalışanları istihdam etme yarışları yaşanır hale gelmiştir. Öyle ki, bazı şirketler kendi aralarında yaptıkları centilmenlik anlaşmalarıyla birbirlerinden çalışan transferi yapmayacakları sözünü vermektedirler. Ayrıca birçok şirkette işten ayrılan çalışan sayısı, yöneticilerin ve departmanların performansını değerlendirken bir performans kriteri olarak kullanılmaktadır. Bu noktada, nitelikli çalışanları kaybetmemek ve sürekli gelişimlerini sağlamak amacıyla doğru bir performans yönetim sistemi uygulamak şirketler için kritik bir konudur.

Performans değerlendirmesi, birden fazla kriter düşünülerek tercih yapmak durumunda kalınan bir karar verme sürecidir ve şirketlerde insan kaynağının verimli bir şekilde yönetimi, çalışanların gelişimi ve organizasyonel performansın artması açısından önemlidir.

1.1 Tezin Amacı ve Kapsamı

Her organizasyonda olduğu gibi yazılım geliştirme ekiplerinin yer aldığı organizasyonlarda da, sürdürülebilir bir başarı elde etmek için bir performans yönetim sistemine ihtiyaç duyulmaktadır. Çalışanların görev ve sorumluluklarının daha keskin bir biçimde ayrıldığı, örneğin üretim yapan bir tesiste, çalışanların performansını değerlendirmek için somut verilerle ölçülebilir kriterler kullanılabilirken, bireysellikten çok takım çalışmasına, teknik becerinin yanı sıra sosyal becerilere de önem veren yeni yazılım geliştirme yaklaşımlarının kullanıldığı organizasyonlarda performans değerlendirmesi yapmak için farklı kriterlere ihtiyaç duyulmaktadır. Bu çalışmada da, bu ihtiyaç doğrultusunda, hızla gelişen teknoloji dünyasına ayak uydurabilmek için Scrum ve Devops yazılım yaklaşımlarını uygulayan yazılım ekiplerindeki performans değerlendirmesinin daha adil ve verimli olması için kullanılabilecek bir performans değerlendirme metodunun adımları anlatılmış ve uluslararası bir firmanın Ar-Ge departmanındaki yazılım test ekibi için uygulaması

yapılmıştır. Çevikliğin ve hızlı uyum sağlamanın öneminin her fırsatta vurgulandığı bilişim dünyasına hizmet eden bir ekibin performans değerlendirme yönteminin de aynı şekilde çevik ve uyarlanabilir olması amacıyla, performans kriterlerinin ağırlıklarını bulanık tamamlanmamış aralık değerli sezgisel tercih ilişkileri ile belirleyen bir yazılım Java programla diliyle geliştirilmiştir. Bulanık tamamlanmamış aralık değerli sezgisel tercih ilişkilerinin kullanılmasıyla, hem karar verici uzmanların tüm ikili karşılaştırmaları yapmak için vakit harcamasındansa sadece bir kriter için ikili karşılaştırmalar yapıp yazılıma verilecek girdinin oluşum sürecinin hızlandırılması, hem de sezgisel bulanık değerler kullanımıyla karar verme sürecindeki emin olamama durumunun etkisinin azaltılması hedeflenmiştir.

1.2 Tezin Yapısı ve Organizasyon

Yapılan bu çalışma 5 ana bölümden oluşmaktadır. Giriş bölümünden sonra yer alan ikinci bölümde, genel olarak kurumlardaki performans yönetim sistemlerinden bahsedilmiş, daha sonra yazılım geliştirilen kurumlarda uygulanan modern yazılım geliştirme yaklaşımları anlatılmış ve son olarak yeni yazılım geliştirme yaklaşımlarının performans yönetim sistemlerine etkisine yer verilmiştir.

Literatür araştırmasının yer aldığı üçüncü bölümde, öncelikle yazılım geliştirilen ekiplerde perfomans ölçümü ile ilgili yapılmış geçmiş çalışmalar özetlenmiş daha sonra da uygulamada da kullanılacak bulanık mantık metotlarıyla performans ölçümü ile ilgili yapılmış çalışmalar incelenmiştir.

Çalışmanın dördüncü bölümünde, uygulamada kullanılacak bulanık mantık, bulanık kümeler, bulanık kümelerin yeni uzantıları, bulanık AHS (Analitik Hiyerarşi Süreci), tercih ilişkileri ve performans gösterge fonksiyonları anlatılmış ve uygulama adımları tek tek açıklanmıştır.

Uygulamanın yer aldığı beşinci bölümde, vaka ve uygulamanın yapıldığı test takımında performansı değerlendirmek için kullanılan kriterler anlatılmış, uygulamaya yardımcı olması için geliştirilen yazılımdan bahsedilmiş ve dördüncü bölümde anlatılan adımların uygulanışına ve her adımda elde edilen verilere yer verilmiştir.

Çalışmanın son bölümünde ise çalışmadan elde edilen çıkarım ve önerilere değinilmiştir.

2. YAZILIM GELİŞTİRME EKİPLERİNDE PERFORMANS YÖNETİMİ

2.1 Performans Yönetimi

Performans Yönetimi, çalışanların bireysel performansını ve ekiplerin yeteneklerini arttırarak işletmelerin sürdürülebilir bir başarı elde etmelerini sağlamak için geliştirilen stratejik ve bütünleşik bir yaklaşımdır (Armstrong ve Baron, 1998).

1900'lü yıllarda Amerika Birleşik Devletleri'nde kamu sektöründe faaliyet gösteren işletmelerde, çalışan perfomanslarının sistematik ve resmi bir şekilde ölçülmesinin ilk örnekleri görülmüştür. Daha sonra, üretimde işbölümü anlayışıyla üretimde bant sistemine geçişin temellerini atan Frederick Winslow Taylor'ın, çalışan verimliliğini ölçmek için geliştirdiği birtakım yöntemler sonucunda, organizasyonlar performans değerlendirme sistemlerini bilimsel olarak kullanmaya başlamışlardır (Uyargil, 2008).

1. Dünya Savaşı'ndan sonra performans değerlendirme kriterleri olarak kişilik özelliklerini temel anlan sistemler geliştirilmiş; 1950'li yıllardan sonra ise, karakter özelliklerinden de kişinin ürettiği iş ya da ortaya çıkardığı sonucu temel alan kriterler kullanılmaya başlanmıştır (Uyargil, 2008). 1960'lı yıllarda, değerlendiren ve değerlendirilen arasındaki mülakatla, çalışanlara kendi performanslarını değerlendirme olanağı verilmiş, 1990'lı yıllarda ise, sadece bir üst yöneticiden değil, çalışanların birlikte çalıştığı diğer kişilerden de alınan geri bildirimlerle 360 derece performans değerlendirme yaklaşımı geliştirilmiştir (Taticchi, 2010).

Türkiye'de Performans Yönetim Sistemi'nin uygulanması öncelikli olarak kamu sektöründe başlamıştır. Daha sonra, işletme biliminin Türkiye'de yaygınlaşması ve modern yönetim metotlarının geliştirilmesiyle birlikte özel sektördeki organizasyonlar da performans yönetimiyle ilgilenmiştir. 2003 yılında yürürlüğe giren, 4857 sayılı İş Kanunu ile birlikte, performans değerlendirme sonuçlarının iş sözleşmelerinin sonlandırılmasında geçerli ve yasal bir sebep olarak kabul edilmesi sonucu özel sektördeki işletmelerin performans yönetimine olan ilgisi daha da artmıştır. Çalışanların performansını değerlendirilmek için kullanılabilecek metotların

sayısındaki artışla beraber işletmeler, en verimli olacak yöntemi seçip uygulama olanağı bulmuşlardır; ancak sadece en uygun yöntemi seçmek, seçilen yöntemin organizasyon yapısıyla uyumlu hale getirilmeden uygulanması sonucu işletmelerin birtakım sorunlar yaşamasına sebep olmuş bu nedenle de işletmeler, organizasyonun diğer sistemleriyle de uyumlu bir şekilde çalışabilecek en verimli yöntemi ve performans yönetim sistemini aramışlardır (Uyargil, 2008).

Performans Yönetimi, bir organizasyondan ve çalışanlarından en iyi sonuçları elde etmek için, planlanmış ve üstünde anlaşılmış hedefler çerçevesinde, takımların performansını anlayarak, yatırım yaparak ve yöneterek kullanılır. Ayrıca, elde edilen sonuçların gerekli iyileştirmeler hakkında bilgi toplamaya nasıl yardımcı olduğunu incelemek için kullanılmalıdır (Giannopoulos, 2015). Bacal (1999), Performans Yönetimi'ni daha geniş bir sistemin içinde çalışan bir sistem olarak analiz etmiştir (Giannopoulos, 2015'te atıfta bulunduğu gibi). Sistem, belirtilen bir amaca ulaşmak için benzer veya birbirine bağımlı şekillerde birlikte çalışan bir dizi bileşen olarak tanımlanır. Bir organizasyon, çeşitli bölümleri birbirine bağlı olarak çalışan bir sistem olarak düşünüldüğünde; performans yönetimi, çalışanların iş olanaklarını ve potansiyelini iyileştirmeyi, çalışma döngüsünü geliştirirken bir yandan şirketin organizasyonunu iyileştirmeyi amaçlayan bir süreç olarak tanımlanabilir (Giannopoulos, 2015).

Performans yönetiminin amacı aşağıdaki gibi listelenebilir:

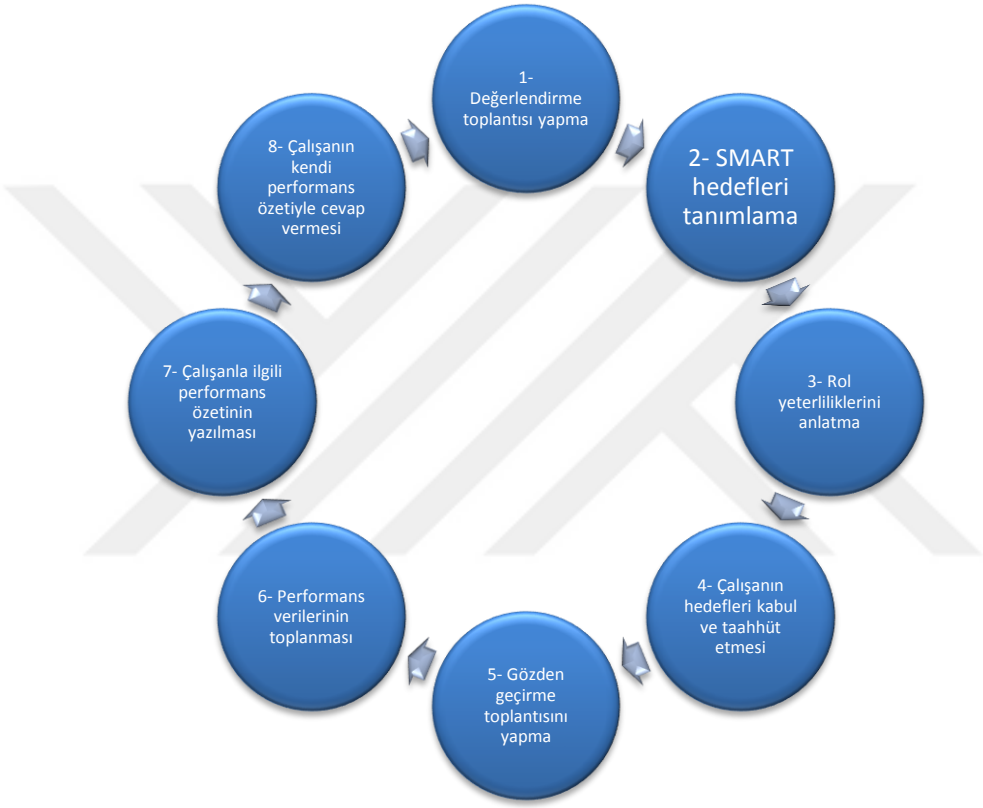
- Çalışanların hedeflerini, organizasyonun hedefleriyle örtüştürmek,
- Çalışanın sahip olması gereken yetenekler, göstermesi gereken davranış biçimi ve rolünün sorumluluğu çerçevesinde neler yapması gerektiği hakkında anlaşmaya varmak ve beklentileri tanımlamak,
- Bireylere yetenek ve yeterliliklerini geliştirip güçlü yönlerini keşfetmeleri için fırsat sunmak,
- Organizasyonun performansını geliştirme amacıyla belirlenmiş bir kişi ve grubun performansını artırmak için değişik pratikleri hayata geçirmek (Armstrong, 2000).

Performans Yönetimi; performans planlama, gözden geçirme ve değerlendirmeyi kapsayan bir şemsiye olarak düşünüldüğünde; Performans Yönetim Sistemi,

yöneticiler tarafından çalışanların işletme içindeki verimlilik ve etkinliklerini değerlendirmek için kullanılan bir araçtır (Arbaiy ve Suradi, 2007).

Çalışanların performansını değerlendirmesi sadece statik bir performans ölçme işi olarak değil de, dinamik bir süreç olarak görüp, performansı planlama, değerlendirme ve geliştirmeyi hedefleyen ve konuyu bütünsel bir yaklaşımla ele alan organizasyonel sistem, Performans Yönetim Sistemi olarak adlandırılmaktadır (Uyargil, 2008).

Performans Yönetim Sistemi Şekil 2.1'deki döngüyle özetlenebilir:



Şekil 2.1 : Performans yönetim sistemi döngüsü.

Performans değerlendirmesi, bir ast ile üstü arasındaki yapılandırılmış resmi bir etkileşim olarak tanımlanabilir (Arbaiy ve Suradi, 2007). Genellikle yıllık periyotta yapılan görüşmelerde, astın geçmiş 12 ay boyunca gösterdiği performans doğrultusunda, güçlü ve zayıf yönleri tartışılır ve gelecek 12 ay için hedefler belirlenir (Trost, 2017 ; Arbaiy ve Suradi, 2007). Birçok şirkette bu döngü, 6 ayda bir yapılan yarıyıl değerlendirmesi ile de desteklenir (Trost, 2017). Değerlendirme gözden geçirme toplantılarında çıkan sonuçlara göre, eğitim ve gelişim ihtiyaçları belirlenir ve ödüller dağıtılır (Al-Heyasi, 2018).

Özetle, bireysel performans çalışanların rolünün gerektirdiklerini ve işinin sorumluluklarını yerine getirmek için sergilediği davranış ve ortaya çıkardığı sonuçların bütünüdür. Organizasyonun performansı ise, paydaşların beklentilerini karşılamak için belirli bir zaman ve ortam için önceden belirlenmiş hedeflere ulaşma başarımıdır. Çalşanların bireysel performansının toplamı organizasyonun performansını belirleyeceğinden, bireysel hedefler organizasyonun hedefleriyle uyumlu olmalıdır.

2.2 Yazılım Geliştirme Yaklaşımları

Proje yaşam döngüsü, projenin başlangıcından kapanışına kadar geçen fazların bütünür (Rose, 2013). Proje yaşam döngüleri sıralandığında, bir ucu öngörülebilir veya plan odaklı yaklaşımlar, diğer ucu ise uyarlanabilir veya değişim odaklı yaklaşımlar olan bir aralıkta sınıflandırılırlar (Rose, 2013).

Öngürülebilir yaşam döngülerinde, ürün ve çıktılar projenin başında tanımlanır ve kapsamdaki herhangi bir değişiklik dikkatlice yönetilir. Uyarlanabilir yaşam döngülerinde ise ürün birden fazla iterasyonda geliştirilir ve her iterasyonun detaylı kapsamı ancak o iterasyon başladığında tanımlanır.

Öngörülebilir yaşam döngüleri (Bütünsel plan odaklı olarak da bilinir), projenin kapsamının, süresinin ve kapsamı teslim etmek için gerekli olan harcamaların, yaşam döngüsünün mümkün olduğunca erken fazlarında belirlendiği döngülerdir.

Yinelemeli ve artırımlı yaşam döngüleri ise proje geliştirilip proje takımının ürünle ilgili bilgisinin arttıkça bilinçli olarak bir veya daha fazla proje aktivitesinin tekrar ettiği fazlardan (İterasyon olarak da bilinir) oluşur. Tekrar eden döngü serileriyle, ürünün işlevselliği birbiri ardına eklenen değişikliklerle arttırılır. Bu yaşam döngüleri ürünü yinelemeli ve artırımlı olarak geliştirir (Rose, 2013).

Uyarlanabilir yaşam döngüleri (Değişim odaklı ya da Çevik metotlar olarak da bilinir), paydaşları düzenli olarak sürece dahil etmeyi ve paydaşlardan gelen geri bildirimlerle gereksinim duyulabilecek büyük değişikliklere hızlı bir şekilde cevap vermeyi amaçlar. Uyarlanabilir metotlar da aynı zamanda yinelemeli ve artırımlıdır; ancak bu metotlarda iterasyonların maliyeti ve süresi sabit, sıklığı fazladır. İterasyonların süresi genelde 2-4 hafta arasındadır.

Uyarlanabilir yaşam döngüsünün kullanıldığı projelerde, her iterasyonda birden fazla işlem gerçekleştirilibilir; ancak genelde ilk fazlarda planlama aktivitelerine odaklanılır. Uyarlanabilir metotlar, özellikle değişimlerin çok sık ve hızlı, gereksinim ve kapsamı önceden tanımlamanın ise zor olduğu ve paydaşlara küçük ama değer katan geliştirmelerin parça parça sunulabileceği alanlarda tercih edilirler (Rose, 2013).

Yazılım geliştirme sektöründe sıklıkla kullanılan öngürülebilir yaşam döngüsü Şelale (Waterfall) metodu iken, uyarlanabilir yaşam döngüsü Agile (Çevik) metodudur.

Aşağıda önce Şelale ve Çevik metotları kısaca açıklanmış, daha sonra günümüzde yazılım ekipleri tarafından sıklıkla tercih edilen ve bir Çevik metot uzantısı olan Scrum metodu ile, çevikliği yazılım geliştirme altyapısına da uyarlamayı amaç edinen DevOps yaklaşımından bahsedilmiştir.

2.2.1 Şelale (Waterfall)

Şelale metodu ismini, yazılım geliştirme aktivitelerinin birinden diğerine basamak basamak geçildiği ve ilerlemenin tek yönlü bir akışla sağlanmasından alır.

Bu yazılım geliştirme metodunda kullanıcı gereksinimlerinin tamamının veya en önemlilerinin projenin başlangıcında yazılım geliştirme ekibine sağlandığı varsayılır. Bu durumda, yazılım geliştirme ekipleri fikir aşamasından uygulamaya kadarki bütün süreçler boyunca sadece ileriye doğru geliştirme yaparlar, geri dönmezler. Yeni bir aşamaya geçilirken bir önceki aşamanın tamamen bittiği kabul edilir.

2.2.2 Çevik (Agile)

1990'lı yılların ortalarında yazılım geliştiriciler ile müşteriler arasında güçlü bir işbirliğinin, ortaya çıkarılan ürünün daha sık teslim edilmesinin ve kendi kendine organize olabilen ekiplere olan ihtiyacın fark edilmesi sonucu bilişim sektöründe Şelale'den daha farklı yazılım geliştirme yaklaşımları uygulanmaya başlamıştır. 2001 yılında, yeni bir yazılım geliştirme yaklaşımına olan ihtiyacı benzer yöntemlerle karşılamaya çalışan 17 deneyimli yazılım geliştiricisi bir araya gelerek bir manifesto yayınlamış ve ilk defa "Çevik" kelimesini kullanmışlardır (Url-4).

Çevik Manifesto'suna göre Çevik bir yazılım geliştirme sürecinin 12 prensibi şöyledir:

1. En önemli öncelik, değerli bir yazılım ürününü erken ve sürekli bir şekilde müşteriye teslim etmektir.

2. Değişiklik istekleri yazılım geliştirme sürecinin ileri aşamalarında bile gelse kabul edilmelidir ve çevik süreçler müşteriye rekabet üstünlüğü sağlayacak şekilde kullanılmalıdır.

3. Yazılım ürünleri, birkaç haftadan birkaç aya kadar değişebilecek bir sıklıkta; ancak mümkün olan en kısa sürede müşteriye sunulmalıdır.

4. Yazılım geliştiricileri ve iş adamları proje boyunca günlük olarak beraber çalışmalıdır.

5. Proje, hevesli bireyler tarafından oluşturulmalı, işin tamamlanması için gerekli ortam sağlanmalı, ihtiyaçlar giderilmeli ve bireylere güvenilmelidir.

6. Bilginin yazılım geliştirme ekibi içerisinde aktarılması için en etkili yöntem yüz yüze iletişimdir.

7. Çalışan bir yazılım ürünü ilerlemenin en temel ölçütüdür.

8. Çevik süreçler sürdürülebilir bir geliştirmeyi destekler. Sponsorlar, yazılım geliştiricileri ve kullanıcılar süresiz olarak sabit bir hızda kalabilmelidir.

9. Teknik mükemmelliğe devamlı dikkat edilmesi ve iyi bir tasarım çevikliği artırır.

10. Sadelik –yapılan işi en aza indirme sanatı- temeldir.

11. En iyi mimari, gereksinim ve tasarım, kendi kendini organize eden ekiplerden ortaya çıkar.

12. Düzenli aralıklarla ekip, nasıl daha verimli olabileceği konusunda kendini incelemeli, gerekli değişiklikleri davranışlarına yansıtmalıdır (Url-3).

Başarılı ekip ve projelerde ortak olarak görülen Çevik yaklaşımın 4 temel değeri şöyledir (Al-Heyasi, 2018):

1. İnsanlar ve aralarındaki etkileşim süreçler ve kullanılan araçlardan daha önemlidir.

2. Belgelendirme değerli olsa da çalışan bir yazılım daha değerlidir.

3. Müşterilerle işbirliği yapmak sözleşme üstüne pazarlık yapmaktan daha önemlidir.

4. Değişikliklere hızlı ve verimli bir şekilde cevap vermek gerekir.

Bir organizasyonun çevik olabilmesi için ise şu özelliklere sahip olması gerekir: Müşteri odaklı, yalın, işbirlikçi, iletişime açık, uyarlanabilir, ölçülebilir, tutarlı, sonuç odaklı ve yansıtıcı (Url-2).

Çevik metodunu uygularken birçok farklı çerçeve kullanılabilir. Bunlardan en sık kullanılanı Scrum iken, diğerleri de Uç Programlama (Extreme Programming), Kanban, Yalın (Lean), Crystal, FDD (Feature Driven Development), SAFe (Scaled Agile Framework), DAD (Disciplined Agile Delivery), LeSS (Large Scale Scrum) gibi sıralanabilir (Url-3, Url-5).

2.2.2.1 Scrum

Çevik Manifestosu'nu hazırlayan 17 yazılım geliştiricisinden 2'si olan Schwaber ve Sutherland, çalıştıkları işletmelerde Scrum ismini verdikleri bir metot uygulamaya başlamıştır ve 2009 yılında uyguladıkları bu yaklaşımı "Scrum Rehberi (The Scrum Guide)" ismini verdikleri bir yayınla paylaşmışlardır. İlk versiyonundan sonra 5 kez revize edilen rehberin son sürümü Kasım 2017'de yayınlanmıştır ve bu çalışmada da son sürümü referans alınmıştır.

Scrum, mümkün olan en yüksek değere sahip ürün ve hizmetleri verimli ve yaratıcı bir biçimde geliştirirken, karmaşık sorunların uyarlamaya açık bir şekilde ele alınabildiği bir çerçevedir (Schwaber ve Sutherland, 2017).

1990'lı yılların başında ürün geliştirme ve yönetme için tasarlanan Scrum, genel olarak aşağıdaki amaçlarla kullanılmaktadır:

1. Mevcut iş pazarlarını, güncel teknolojileri ve ürün yeterliliklerini araştırma ve tanımlama

2. Ürün ve ürünlerle ilgili geliştirme yapma

3. Ürün ve ürünlerle ilgili geliştirmelerin yeni bir sürüm olarak bir günde birden fazla çıkartılması

4. Bulut (çevrimiçi, güvenli, isteğe bağlı) ve diğer iş ortamların geliştirilmesi ve sürdürülmesi

5. Geliştirilen ürünlerin sürdürülebilir olması ve yenileme. (Schwaber ve Sutherland, 2017)

En çevik yazılım geliştirme metotlarından biri olan Scrum'ın 18 milyondan fazla yazılım geliştirme çalışanı tarafından her gün uygulandığı düşünülmektedir (Url-1).

Scrum, günlük hayatta toplum tarafından kullanılan, donanım, yazılım, gömülü yazılım, otonom araçlar, hükümet sistemleri, okul sistemleri, pazarlama,

organizasyonların yönetimi gibi birçok alanda geliştirme yapmak için kullanılmaktadır. Bilişim teknolojileri pazarı geliştikçe ve bu pazardaki etkileşimler hızlanarak arttıkça ortaya çıkan karmaşıklıkla Scrum uygulayarak mücadele edilebileceği kanıtlanmıştır (Schwaber ve Sutherland, 2017).

Scrum'ın özellikle, iteratif ve artımlı bilgi aktarımında verimli olduğu ortaya konmuş ve günümüzde, hizmet, ürün ve organizasyon yönetimi için oldukça yaygın olarak kullanılır hale gelmiştir (Schwaber ve Sutherland, 2017).

Scrum'ın temelinde küçük ekipler vardır. Bu ekipler oldukça esnek bir yapıya sahiptir ve uyarlanabilirler. Bu güçlü yönleri sayesinde, çok fazla kişinin ürün ve hizmet geliştirdiği, yaygınlaştırdığı, işlettiği ve idame ettirdiği farklı takım yapılarında (tek ekip, birkaç ekip, çok sayıda ekip ve ekip ağlarında) da Scrum uygulanabilir. Scrum'ın temelindeki sürecin deneysel kontrolü teorisi (veya deneycilik) bulunur. Deneyciliğe göre bilgi, tecrübeyle ve öğrenilen şeylere dayanarak verilen kararlarla elde edilir. Ögörülebilirliği en üst seviyeye çıkarmak ve riski kontrol edebilmek için Scrum'da yinelemeli ve artımlı bir metot kullanılır (Schwaber ve Sutherland, 2017).

2.2.2.2 DevOps

DevOps, İngilizce Development (Geliştirme) ve Operations (İşletme, çalıştırma) kelimelerinin birleştirilmesiyle türetilmiştir. Yazılım geliştirme işlemlerinin başlangıcından, geliştirilen ürünün kullanıma sunulduğu ortama konulup faaliyete geçirilmesine kadarki süreçteki işlemlerin tümünü kapsar ve geliştirme ve çalıştırma ekipleri arasındaki işbirliğini geliştirmeyi amaçlar (Hüttermann, 2012).

DevOps, çevik yaşam döngüsü uygulayan yazılım ekiplerinin yaşadıkları sıkıntılar doğrultusunda bir "Çevik Altyapısı" olarak, ekip çalışanlarına daha esnek bir altyapı sunabilmek amacıyla ortaya çıkmış, daha sonraları bilişim sektöründe, sadece altyapı için değil, daha geniş kapsamda devrimsel bir harekete dönüşmüştür (Url-1).

DevOps, yazılım geliştiriciler ve bilişim operasyonu profesyonelleri arasında iletişim, işbirliği, entegrasyon ve otomasyonu vurgulayan kültürel ve profesyonel bir harekettir ve 5 temel prensibi aşağıdaki gibidir (Hüttermann, 2012; Krishna Kaiser, 2018):

1. Kültür: Çalışanlar sadece kendi çalıştıkları iş parçacığının değil, tüm ürünün sorumluluğunu üstlenmeli, konfor alanının dışına çıkıp yaratıcı düşünceye sahip olmalıdır. İstenildiği kadar deney yapılabilir; çünkü başarısız olunduğunda geri dönüş

yapmak otomasyon sayesinde oldukça kolaydır. İletişim, işbirliği ve sürece dahil olmuş kişilerle yakın temasta bulunmak esastır.

2. Otomasyon: Hızlı geri bildirim alabilmek ve hızlı teslimat yapabilmek için otomasyon DevOps'un temeldir.

3. Yalın: Bu prensip, işleri basit tutup fazla karmaşıklaştırmadan çalışmaya devam etmeye yardımcı olur.

4. Ölçme: Yazılımın genel durumunu görüp, geri bildirimde bulunabilmek ve önlem alabilmek için ölçüm yapmak esastır.

5. Paylaşma: DevOps, insanların fikirleri, süreçleri ve araçları paylaştığı bir kültür oluşturur. Yazılm teslim süresinin hızlandırılmasının hedeflendiği bir durumda, aynı ürün veya servisle ilgili çalışan herkesi tek bir ekibin içinde konumlandırmak ve bilgi paylaşımını teşvik etmek gerekir. Bu, rekabet ve şüphecilikten çok işbirliğine yol açar.

Yazılım geliştiricileri ve operasyon ekipleri arasındaki iş akışını iyileştiren DevOps, ortak sorumluluk kavramını aşılamayı amaçlar. Geliştirilmiş iş akışı, kısaltılmış geri bildirim döngüleri, paylaşılan pratikler ve otomasyon, bilişim sektöründeki tedarik zincirinin üretim ve değer katma hızını arttırmasına yardımcı olur (Url-2).

DevOps'un süreçleri aşağıdaki gibidir (Krishna Kaiser, 2018):

• Sürekli entegrasyon (Continuous Integration): Bu süreç, farklı kod parçacıkları üstünde çalışan yazılım geliştiricilerin kodlarının en az günde bir kere araçlar yardımıyla otomatik olarak birleştirilmesini ve statik kod kontrolünü ve birim testlerin çalıştırılmasını kapsar.

• Sürekli teslimat (Continuous delivery): Birleştirilmiş kod üstünde entegrasyon, system ve kullanıcı kabul testlerinin koşulup, yazılımın canlı ortama konulmasına kadarki süreci kapsar.

• Sürekli yerleştirme (Continuous deployment): Bu süreç, sürekli teslimat sürecindeki tüm controller başarılı olduktan sonra, yazılımın otomatik olan canlı müşteri ortamına yerleştirilmesi işlerini kapsar.

DevOps sürecinde temel olarak aşağıdaki adımlar işletilir (Url-5):

1. Geliştirilen kod, ortak kaynak kontrol sistemine konur.

2. Tüm kod, derlenmek ve birleştirilmek için kaynak kontrol sisteminden çekilir.

3. Birim testler otomatik olarak çalıştırılır. Testler başarısızsa 1. adıma geri dönülür, başarılıysa 4. adımdan devam edilir.

4. Otomatik bir yazılım aktarım aracı kullanılarak test ortamına yazılımın dağıtımı yapılır.

5. Sürekli entegrasyon (continuous integration) sunucusu, yeni yazılım sürümünü oluşturur.

6. Entegrasyon, sistem ve kullanıcı kabul testleri yapılır. Testler başarısızsa 1. adıma geri dönülür, başarılıysa 7. adımdan devam edilir.

7. Başarılı sürümler depolanarak saklanır.

8. Otomatik bir yazılım aktarım aracı kullanılarak müşteri ortamına yazılımın dağıtımı yapılır.

9. Veritabanları güncellenir.

10. Uygulamalar güncellenir.

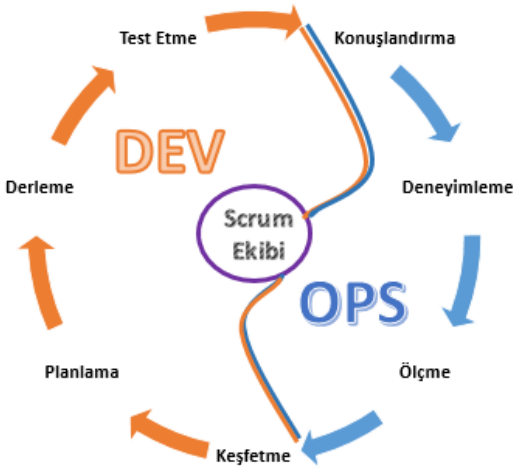
11. Uygulama ve ağ performansı izlenir ve sorunlar oluşmadan engellenmeye çalışılır.

Scrum, karmaşık projeleri yönetmek için mükemmel bir çerçeve olsa da organizasyonların çevikliğini arttırmak için yetersiz kalır (Url-2). Bu noktada, organizasyonların daha çevik olabilmesi için DevOps hareketi yardımcı olur.

DevOps ve Scrum döngüsü Şekil 2.2'deki gibi gösterilebilir (West ve Groll, 2007).

2.3 Yazılım Geliştirme ve Performans Yönetimi

İşletmelerde, ekibin bütününe değil de çalışanlara bireysel olarak ödeme yapıldığından, Performans Yönetim Sistemleri çalışanların bireysel performanslarını değerlendirmek üzere kurulmuştur. Günümüzde, yazılım ekibi yöneticileri de işletmelerin Performans Yönetim Sistemi'ne uyum sağlamak için çalışanların performanslarını bireysel olarak değerlendirmektedir. Bu yaklaşım, her çalışanın görev ve sorumluluklarının net ve ayrıştırılabilir olduğu geleneksel üretim işletmelerinde daha kolay uygulansa da projelerin farklı fazlarında farklı görevler alan yazılım geliştirme ekibi çalışanlarının sorumluluklarını ayrıştırmak ve performanslarını ölçmek daha zordur. Bu nedenle, yazılım geliştirme ekiplerinde performans yönetimi daha farklı ele alınmaktadır (Narayan, 1996).



Şekil 2.2 : DevOps ve Scrum döngüsü.

2.3.1 Çevik sistemlerde bireysel performansın ölçülmesi

Çevik yazılım geliştirmede kurallardan çok prensipler vardır. Çevik metotları kullanan yazılım ekipleri takımın ve bireylerin performansını ölçmekte zorlanırlar; çünkü çevik takımların nasıl bir performans sergilediğini gösteren bir kriter yoktur (Al-Heyasi, 2018).

Yazılım geliştirme ekibindeki çalışanlar, farklı geçmişlere ve farklı rollere sahip olduğundan çevik ekiplerdeki "iyi" tanımı her bir çalışan için farklı olabilir:

• Yazılım geliştiricisi "iyi"yi güzel tasarlanmış, çalışan bir yazılım olarak düşünebilir.

• Ürün sahibi için "iyi", mümkün olduğunca çok özellik teslim etmek olabilir.

• Bir proje yöneticisi için ise "iyi", işi zamanında ve bütçe dahilinde tamamlamak olarak görebilir. (Al-Heyasi, 2018)

Davis (2015)' e göre, çevik sistemler müşteriler ve ürün geliştiriciler arasındaki işbirliğine dayandığından, ürünün geliştirilmesi sırasındaki verileri, performans kriterine çevirerek her iki taraf arasındaki iletişimi geliştirmek esastır (Al-Heyasi, 2018'de atıfta bulunulduğu gibi).

Gamble ve Hale (2013) çevik yazılım geliştirme ekiplerindeki bireysel performansı simgeleyebilecek 4 kriteri aşağıdaki tanımlamıştır:

1. Sprint toplantılarındaki katkı

2. Ekibin ilerlemesindeki bireysel etki

3. Ortaya çıkan yazılımın kalitesindeki bireysel etki

4. Projenin başarılı bir şekilde tamamlanması sürecinde takım arkadaşlarının kişinin performansıyla ilgili izlenimi

2.3.2 DevOps/Scrum ekibi çalışanlarının sahip olması gereken özellikler

Daha önce de bahsedildiği gibi, DevOps ve Scrum uygulanan projelerin en önemli özelliği değişime ve esnekliğe açık olmasıdır. Dolayısıyla, DevOps ve Scrum ekibi çalışanlarının da bu doğrultuda bazı özelliklere sahip olmaları beklenir.

Scrum teorisine göre Scrum ekipleri çapraz fonksiyonlu (cross-functional) olmalıdır ve çapraz fonksiyonlu ekipler, sadece ekip içindeki çalışanlarla, yani dışarıdaki kişilere bir bağımlılık gerektirmeden, görevleri bitirebilecek yeteneğe sahiptir (Schwaber ve Sutherland, 2017).

Ekip elemanlarının sahip olması gereken karakteristik özellikler şöyle özetlenebilir: kararlı, güvenilir, güçlendirilmiş, hevesli, sorumluluk sahibi, odaklanmış, iş merkezli, iletişime açık, kalite odaklı (Url-2).

3. LİTERATÜR ARAŞTIRMASI

3.1 Yazılım Geliştirme Ekiplerinde Performans Ölçümü İle İlgili Yapılan Çalışmalar

Yazılım geliştirme ekiplerinde performans ölçümü ile ilgili yapılan çalışmaların bir kısmı kullanılan performans kriterlerini incelerken, bir kısmı da kriterlerin kullanımını, yani değerlendirme metotlarını araştırmıştır.

Kanij ve diğ. (2014), yazılım test mühendislerinin performans değerlendirmesi yapılırken kullanılan kriterlerin birbirlerine göre önemlerini, literatür taraması ve sektör çalışanlarıyla anket yaparak araştırmıştır. 2012 yılında yaptıkları çalışma, "Bulunan hata sayısı"nın en az önemli, "Hata Raporlama Kalitesi"nin ise en önemli kriter olduğunu ortaya koymuştur (Kanij ve diğ, 2012). Bir sonraki yıl devam eden araştırmalarında, Küme ve Kartopu Örnekleme metoduyla seçtikleri 18 proje yöneticisiyle yapılan web tabanlı bir anket çalışmasıyla önerdikleri yaklaşımı yenilemiş ve "Proje Tesliminden Sonra Bulunan Hata Sayısı" ve "Test Senaryolarını Koşma Verimliliği" kriterlerinin daha önemli olduğunu savunmuşlardır (Kanij ve diğ, 2014).

Alnaji ve Salameh (2015), Scrum yazılm geliştirme metodunu kullanan ekiplerdeki bireylerin performasını ölçmek için bir çerçeve tasarlamış ve 5 farklı kriter belirlemiştir: Üretkenlik, Verimlilik, Sosyal Yetenekler, Mentorluk ve Ekip İçi İşbirliği ve Bilgi Genişliği.

Podjavo ve Berzisa (2017)'nın, yazılım geliştirme projelerinde ekip performansını ölçmek için hazırladıkları kılavuz, niteliksel ve öznel kriterlerle ve onları etkileyen teknik ve sosyal faktörleri içermektedir ve takımdaki iletişim ve işbirliğini değerlendirmek için sosyometrik metodunu önerir.

Yang ve diğ. (2009), işletmelerin ve proje yöneticilerinin, çalışanların mevcut yeterliliklerini anlayıp daha iyi bir ekip oluşturma süreçlerine yardımcı olabilmek için, Analitik Ağ Süreci (AAS) metodunu kullanarak çok kriterli bir değerlendirme modeli

oluşturmuşlardır. Bu modelde, 3 farklı kategoride (Proje Planlama, Proje Yürütme ve Kontrol ve Bilgi Sistemi Oluşturma) belirlenen 65 kriter, her kategorideki stratejik önemlerine göre ağırlıklandırılmış ve kriter göstergeleri elde edilmiştir.

Chen, yazılım geliştiricilerinin performansını değerlendirmek için Fonksiyonel Nokta Analizi (FNA)'ni kullanarak bir uygulama yapmıştır. FNA yöntemini kullanmanın değerlendirme sürecindeki nesnellik ve adaleti arttırdığını savunmuş ve proje boyunca tamamlanacak fonksiyonların geliştirilmesi için gereken iş yükünün Fonksiyonel Nokta ölçeğinde ifade edilip, çalışanların tamamladıkları işin sayısallaştırıldığı bir performans değerlendirme uygulaması yapmıştır. Chen bu yöntemle, geleneksel performans değerlendirme yöntemlerindeki öznel faktörlerin sebep olduğu sapmaların ve kısıtlarının aşıldığını ifade etse de; bu yöntemle elde edilen değerlendirme sonuçlarının, yazılım geliştiricilerinin etik, karakter ve çalışma davranışları gibi bireysel özelliklerinin değerlendirilmesiyle birleştirilmesi gerektiğini eklemiştir (Chen, 2008). Ancak bunun için bir yöntem önermemiştir.

Raza ve diğ. (2017), yazılım mühendisliği öğrencilerinin performans analizini ve geliştirme önerilerini otomatik bir şekilde yapan bir araç geliştirmişlerdir. ProcessPAir adını verdikleri bu araç, tanımlama, ayarlama ve analiz etme adımlarını içerir. Tanımlama aşamasında, süreç uzmanları, incelenen geliştirme sürecine uygun olan performans modelini tanımlayıp, üst ve ast seviye performans göstergelerini hiyerarşik olarak sebep-sonuç ilişkisine göre düzenlerler. Ayarlama aşamasında performans modeli, her bir performans göstergesinin istatistiksel dağılımı, girilen veri setindeki performans göstegelerinin kendi aralarındaki ilişki ve birçok süreç kullanıcısının performans verisine göre otomatik olarak kalibre edilir. Analiz aşamasında ise her bir yazılım geliştiricisinin performans verisi, performans problemlerini (üst seviye performans göstergeleri) ve olası kök sebeplerini (alt seviye performans göstergeleri) tanılamak ve sıralamak için ProcessPAIR tarafından otomatik olarak incelenir (Raza ve diğ, 2017).

Alahyari ve diğ. (2018) İsveç'teki bir bilgi teknolojileri şirketinde ve bu şirketin bir müşterisinde, anket ve mülakatlarla yaptıkkları araştırma sonucunda çevik takımlarda başarıyı getiren en önemli 5 faktörün takım içerisindeki iyi iletişim ve işbirliği, güvenilir bir takım ortamı, takım içerisindeki arkadaşça ve pozitif ortam, çalışanların kendilerini geliştirme istek ve şansına sahip olmaları ve takım üyeleri arasındaki zihinsel anlaşma olduğunu savunmuşlardır.

Parizi ve diğ. (2019), yazılım ekibi çalışanlarının performansını değerlendirmek için Git tabanlı bir mimari yapı geliştirerek çalışanların gün boyu proje için harcadıkları zamanı, yazdıkları kod satır sayısını ve değişiklik yaptıkları dosya sayısını ölçmüşler; ancak çalışılan projenin zorluğu ya da projede kullanılan yeni yazılım yaklaşımlarının performansa etkisini araştırmamışlardır. Ayrıca değerlendirilen kriterlerin ağırlıkları yazarların kendi deneyimleri ve uzman görüşleriyle belirlenmiş ancak çalışmada bu sürecin metodolojik detaylarına yer verilmemiştir.

3.2 Bulanık Metotlarla Performans Ölçümü İle İlgili Yapılan Çalışmalar

Chandran ve Kandasamy (2011), performasın somut özelliklerini ve bir çalışanın yapılandırılmış bir değerlendirme sistemindeki çıktısının nicel ölçümünü hesaba katan bulanık bir performans değerlendirme sistemi önermişlerdir. Önerilen bu sistemin uygulama prosedürü, yöneticilerden değerlendirmeleri toplama, bunları bulanık sayılara dönüştürme ve çalışanın performans skorunu türetme aşamalarından oluşur. Bulanık sayıları sıralamak için puantaj (Scoring) metodu kullanılmıştır. Önerilen bulanık performans değerlendirme sistemi hedef odaklı, yapılandırılmış, bilgilendirici ve rasyoneldir. Bu sistem, işletmenin her bir farklı hedefi için çalışanların yaptıkları katkıyı gözlemleyip kaydeder. Kaydedilen bu bilgi, yöneticiler için oldukça faydalı bir bilgidir. Chandran ve Kandasamy (2011) önerdikleri sistemin güçlü yanının bu olduğunu savunmuştur.

Arbaiy ve Suradi (2007), çalışanların performans değerlendirme sistemini bulanık mantık yaklaşımıyla incelemişlerdir. Yaptıkları çalışmada, performans değerlendirmesi için işin başarımı, bilgi ve beceri, kişisel özellikler ve toplum hizmeti şeklinde 4 kategorinin incelenmesi gerektiğinden hiyerarşik bulanık çıkarım yaklaşımını kullanmayı uygun bulmuşlardır. Performans ölçme sistemindeki belirsiz veri, etiket ve güven değerinden oluşan bulanık değerlerle eşleştirilir. Çalışmanın çıktısı, çalışanların performans sıralamasını verir (Arbaiy ve Suradi, 2007).

Shaout ve Khalid Yousif (2014) da bulanık mantık kullanarak bir çalışan performansı değerlendirme sistemi tasarlamış ve bu sistemi Sudan'daki bazı petrol şirketi çalışanlarının performansını değerlendirerek uygulamışlardır. Tasarlanan sistemde, performans değerlendirme modüllerinin normal işlemlerinin yanı sıra adım adım çıkarım işlemleri de dahil edilmiştir. Bu işlemler, ilişkilerin kompozisyonunda ve min operatörü, cebirsel ürün, sup-min ve sup-ürün gibi birleştirme yöntemlerinde çeşitli

hesaplama ayrıntılarını gösterir. Sistem, çeşitli benzerlik kriterleri kullanarak nihai sonucu analiz etmek ve raporlamak için ilave analiz modülüne temel oluşturur. Verileri korumak, çıkarım mantığını oluşturmak ve kullanıcı arayüzlerini geliştirmek için MS Access veritabanı kullanılmıştır (Shaout ve Khalid Yousif, 2014).

Ahmed ve diğ. (2013) de bulanık mantık kullanarak çeşitli performans değerlendirme kriterlerini gözeten bir performans değerlendirme sistemi önermişlerdir. Önerilen yaklaşımdaki ana amaç, çalışanların performans endekslerinin çeşitli nitel ve nicel değerlendirme kriterlerindeki başarımları göz önünde bulundurularak belirlenip, en yüksek performans endeksine sahip çalışanın seçilmesidir. Bulanık kontrol, seçilen kriterlerdeki başarım sonuçlarını birleştirerek genel performans endeksini belirlemek için kullanılır ve çalışanın performans değerlendirme hesaplaması sırasında rahatlık sağlayacak sayısal değerler sunar. Bu çalışmada bir şirketteki en iyi çalışanların geçmiş performans verileri kullanılmış ve bulanık mantığın karmaşık hesaplamaları MATLAB yazılımıyla yapılmıştır (Ahmed ve diğ, 2013).

Literatürde bulanık AHS yöntemiyle performans kriterlerini önceliklendirme için birçok çalışma yapılmıştır. Bunlardan biri de Chen ve diğ. (2014) Vietnam'daki üniversitelerdeki eğitmenlerin performansını değerlendirmek için 17 karar vericiyle 6 ana ve 14 alt performans kriterinin ağırlıklarını hesaplayarak bir çalışma yapmışlardır. Bir başka çalışma da Bozbura ve diğ. (2007) 'nin 5 ana ve 20 alt kriterle insan sermayesini değerlendirmek için kullanılan kriterlerin önceliklendirilmesi için yapılmıştır.

Yousif ve Shaout (2016), Sudan üniversitelerindeki akademik personelin performans değerlendirmesini yapmak için bulanık AHS yöntemiyle kriter ağırlıklarını belirlemiş, TOPSIS yöntemiyle de üniversiteler arasındaki sıralamayı yapmışlardır.

Bulanık AHS ve TOPSIS metodlarının birlikte kullanıldığı bir başka uygulama da Kusumawardani ve Agintiara (2015) 'nın bir telekomünikasyon şirketinin birçok bölgesindeki insan kaynakları yönetici adayları arasında bir değerlendirme yaptıkları çalışmadır. Çalışma sonucunda, birçok bölgede göreve getirilen kişiyle bulanık AHP-TOPSIS metotlarının sonucunda önerilen kişinin örtüştüğü görülmüştür.

4. METODOLOJİ

1965 yılında L.A. Zadeh tarafından önerilen Bulanık Mantık, doğru ya da yanlış gibi kesin ifadeler yerine kademeli ya da nitelikli ifadeler kullanan bir mantık türüdür ve insanın zihinsel süreçlerinden doğan belirsizlikleri modellemek üzere ortaya atılmıştır (Zadeh, 1984; Memik, 2017).

Doğrunun "1", yanlışın "0" ile ifade edildiği klasik mantık, gerçek hayattaki durumların ifade edilmesinde yetersiz kalır. Bulanık akıl yürütme sonrasında çıkan sonuçlar, klasik mantıkla ortaya çıkan sonuçlar kadar kesin değildir; ancak daha geniş bir söylem alanını kapsarlar (Zadeh, 1984).

Uysal (2010)'un da belirttiği gibi günlük hayatta alınan kararlar insan zihninin karmaşık süreçlerinden geçtiği için birçok belirsizlik içerir (Memik, 2017). Bulanık mantık sayesinde, "oldukça uzun" veya "çok hızlı" gibi kavramlar, bilgisayar programlamada insanın düşünme yapısına daha yakın bir şekilde matematiksel olarak formüle edilebilir ve bilgisayarlar tarafından işlenebilir (Chandran ve Kandasamy, 2011). Yani bulanık mantık sayesinde insanın düşünce yapısındaki belirsizlikler de dikkate alınarak nitel kavramlara nicel bir karşılık belirlenebilir.

Klasik mantıkla oluşturulan sistemlerde verilerin tam ve kesin olması gerektiğinden , karmaşıklık ve belirsizlik içeren yöntemleri modellemek zordur. Bulanık mantık ise, insanların kesin olmayan ifadeler ile düşünme şekliyle örtüşen bir mantık sistemidir ve bu nedenle bulanık mantığın kullanıldığı modellerin daha başarılı sonuçlar ortaya çıkardığı savunulmuştur (Zadeh, 1984; Memik, 2017). Ross (1999), bulanık mantığa dayalı sistemlerin kullanımının oldukça elverişli olduğu durumların, davranışları tam olarak bilinemeyen fazlaca karmaşıklık içeren sistemlerin modellenmesi ve hızlı sonuçlar elde edilmesi istenen problemlerin çözümlenmesi olduğunu belirtmiştir.

Zadeh (1984), teorisinde kullanılan terimleri özetle şu şekilde tanımlamıştır:

• Bulanık dönüştürücüler: Bulanık kümenin üyelik fonksiyonunu, tam üyelik ve üye olmama arasındaki geçişi genişleterek, bu geçişi keskinleştirerek ya da geçiş bölgesinin konumunu hareket ettirerek değiştiren operasyonlardır.

• Bulanık kümeler: Net bir şekilde tanımlanmış bir üyeliği olmayan, fakat nesnelerin 0'dan 1'e kadar üyelik dereceleri almasına izin veren kümelerdir.

• Dilsel değişkenler: Belirli bir problemde belirli bir bulanık kümeyi temsil etmek için kullanılan, "büyük", "küçük", "orta" ya da "uygun" gibi sıradan dil terimleridir.

• Ultra bulanık kümeler: Üyelik fonksiyonun kendisinin de bir bulanık küme olduğu bulanık kümelerdir. Böylece kümedeki bir nesneye 0 ile 1 arasında bir üyelik notu vermek yerine olası bir üyelik derecesi aralağı atanır. Örneğin 0.55 yerine 0.4 – 0.6 arasında denir.

4.1 Bulanık Kümeler

Zimmerman'a (1992) göre bulanık küme teorisi, belirsiz bir karar ortamında katı bir matematiksel çerçeve sağlar. Sugeno ve Kang (1986) bulanık kümelerin, bulanık ilişkiler ve kriterlerin olduğu durumlar için uygun bir modelleme dili olarak da kabul edilebileceğini söylemişlerdir.

Klasik kümelerde temel mantık, kümeye üyelik yani ait olma durumudur. Eğer bir nesne bir kümeye aitse o kümenin elemanı olarak kabul edilir, ait değilse o kümenin elemanı değildir. Aitlik durumu "1", ait olmama durumu da "0" ile ifade edilir.

Bulanık küme elemanları için ise "1" ve "0" gibi kesin değerlerin yanı sıra, esneklik sağlayan ara değerlerle de ait olma ve olmama durumları ifade edilir. "0" ile "1" arasındaki değerler farklı üyelik derecelerini göstermektedir. Bulanık küme kavramına göre, bir eleman bir kümeye üyeliğinin tam olmasına gerek duymadan başka bir kümenin de elemanı olabilmekte ve kümeler arasında kademeli geçişler olabilmektedir.

X evrenindeki A bulanık kümesi aşağıdaki gibi tanımlanmıştır:

$$A = \{(x,\ \mu_A(x)) \mid x \in X\}, \qquad \mu_A : X \rightarrow [0,1] \tag{4.1}$$

$\mu_A(x)$, x elemanının A kümesine üyelik derecesini gösterir. Eğer $\mu_A(x)$ = 1 ise, x elemanı A kümesine tam olarak aittir. $\mu_A(x)$= 0 ise, x elemanı A kümesine ait değildir.

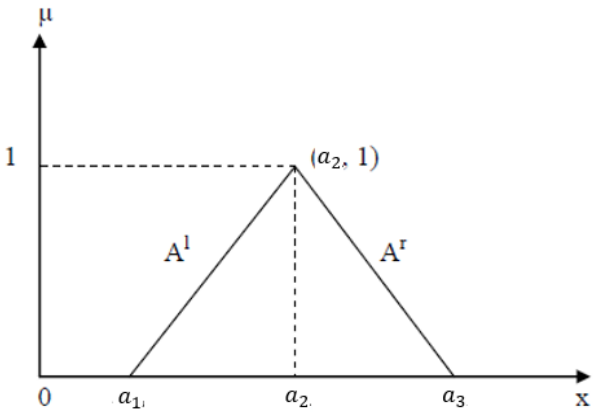
Çelikyılmaz ve Türksen (2015), üyelik fonksiyonları genellikle düz çizgiler ile formülize edildiğini belirtir ve gerçek hayat uygulamalarında üyelik değerleri dağınık noktalar halinde şekillense de bu değerler normalize edilerek üçgensel, yamuksal veya Gaussian üyelik fonksiyonları olarak ideal hallerine kavuşmaktadırlar.

4.1.1 Üçgensel üyelik fonksiyonları

Üçgensel üyelik fonksiyonu a_1, a_2 ve a_3 ile ifade edilen üç parametre ile aşağıdaki gibi tanımlanmıştır (Gök, 2015):

$$\mu_A(x, a_1, a_2, a_3) = \begin{cases} \dfrac{x - a_1}{a_2 - a_1} & , \ a_1 \leq x \leq a_2 \text{ ise} \\ \dfrac{a_3 - x}{a_3 - a_2} & , \ a_2 \leq x \leq a_3 \text{ ise} \\ 0 & , \ x < a_1 \text{ veya } x > a_3 \text{ ise} \end{cases} \tag{4.2}$$

Şekil 4.1'de a_1, a_2 ve a_3 parametrelerinden oluşan [a_1, a_3] aralığında tepe noktası (a_2,1) olan üçgensel üyelik fonksiyonuna ait grafik gösterilmektedir. Uygulamalarda a_2 parametresi sıklıkla orta noktayı ($a_2 = (a_1+a_3)/2$) temsil etmekte ve merkezi üçgensel bulanık sayı olarak ifade edilmektedir. Bojadziev ve Bojadziev (2007) 'in belirttiği gibi Modelleme ve hesaplama kolaylığı sağlaması bakımından uygulamada A_l (sol) ve A_r (sağ) aralığına ve a_2 (A_m) orta değerine sahip üçgensel üyelik fonksiyonlarından faydalanılmaktadır (Gök, 2015'te atıfta bulunulduğu gibi).



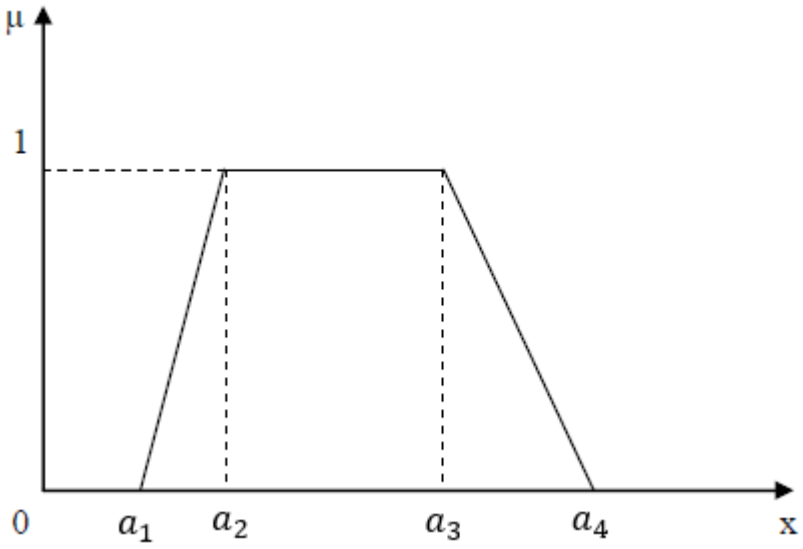
Şekil 4.1 : Üçgensel üyelik fonksiyonu grafiği.

4.1.2 Yamuksal üyelik fonksiyonları

Üçgensel Yamuksal üyelik fonksiyonu a_1, a_2, a_3 ve a_4 ile ifade edilen dört parametre ile aşağıdaki gibi tanımlanmıştır (Gök, 2015):

$$\mu_A(x, a_1, a_2, a_3, a_4) = \begin{cases} 0 & , \quad x < a_1 \text{ ise} \\ \dfrac{x - a_1}{a_2 - a_1} & , \quad a_1 \leq x \leq a_2 \text{ ise} \\ 1 & , \quad a_2 \leq x \leq a_3 \text{ ise} \\ \dfrac{a_4 - x}{a_4 - a_3} & , \quad a_3 \leq x \leq a_3 \text{ ise} \\ 0 & , \quad x > a_4 \text{ ise} \end{cases} \tag{4.3}$$

$a_1 < a_2 \leq a_3 < a_4$ parametrelerine sahip yamuksal üyelik fonksiyonuna ait grafik Şekil 4.2'deki gibidir (Gök, 2015):



Şekil 4.2 : Yamuksal üyelik fonksiyonu grafiği.

Bojadziev ve Bojadziev (2007) da belirttiği gibi, eğer yamuksal üyelik fonksiyonu parametrelerinden ikinci ve üçüncü parametreler birbirine eşitse ($a_2 = a_3$), yamuksal üyelik fonksiyonu üçgensel hale dönüşmektedir (Gök, 2015'te atıfta bulunulduğu gibi).

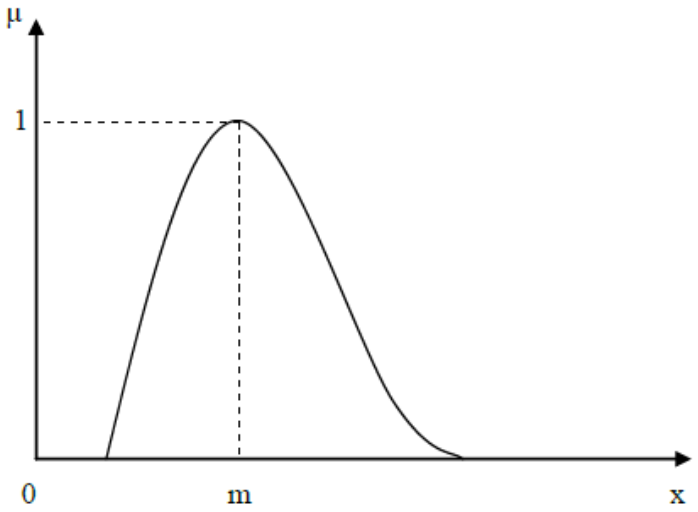
4.1.3 Gaussian üyelik fonksiyonları

Gaussian üyelik fonksiyonu m ve σ olmak üzere iki parametre ile aşağıdaki gibi ifade edilmektedir. m değeri üyelik fonksiyonunun merkezini, σ ise genişliğini gösterir. Yen ve Langari (1999)'nin belirttiği gibi σ değerinin değişmesiyle fonksiyonun şekli de

değişmektedir. değeri küçüldükçe üyelik fonksiyonunun biçimi dar olurken, değer büyüdükçe üyelik fonksiyonu yassı bir hal alır (Gök, 2015'te atıfta bulunulduğu gibi).

$$\mu_A(x, m, \sigma) = e^{\frac{-(x-m)^2}{2\sigma^2}} \tag{4.4}$$

Şekil 4.3'te Gaussian üyelik fonksiyonuna ait grafik gösterilmiştir. Şen (2003)'in belirttiği gibi üyelik derecesinin 1'e eşit olduğu noktayı m değeri gösterirken, σ parametresi ise m değeri etrafındaki sapmaların bir ölçütünü ifade etmektedir (Gök, 2015'te atıfta bulunulduğu gibi).



Şekil 4.3 : Gaussian üyelik fonksiyonu grafiği.

4.2 Bulanık Kümelerin Uzantıları

4.2.1 Aralık değerli bulanık kümeler

Aralık değerli bulanık kümeler, elemanların üyelik derecelerinin, [0,1] aralığındaki gerçek sayılarla ifade edildiği bulanık kümelerdir (Lee, 2004).

$[I]$'nın [0,1] aralığındaki kapalı alt aralıklar kümesi ve μ^l alt uç, μ^r üst ucu simgelerken $\mu = [\mu^l, \mu^r] \in [0,1]$ olduğu durumda, aralık değerli A bulanık kümesi, aşağıdaki üyelik fonksiyonuyla tanımlanır:

$$A = \{(x,\ \mu_A(x)) \mid x \in X\}, \qquad \mu_A : X \rightarrow [I] \tag{4.5}$$

x elemanının A kümesine üyelik derecesini gösteren $\mu_A(x) = [\mu_A^l(x), \mu_A^r(x)]$, aralık değerli bulanık bir sayıdır.

A ve B 'nin aralık değerli bulanık kümeler olduğunu varsayarsak, x elemanının üyelik derecesi aşağıdaki gibi ifade edilir:

$$\mu_A(x) = \left[\mu_A^l(x), \mu_A^r(x)\right]$$
$$\mu_B(x) = \left[\mu_B^l(x), \mu_B^r(x)\right] \quad (4.6)$$

Aralık değerli bulanık kümelerin temel işlemleri aşağıdaki gibi tanımlanmıştır:

$$A \cup B = \left\{\left(x,\ \mu_{A\cup B}(x)\right) \mid x \in X\right\}$$
$$\mu_{A\cup B}(x) = \left[\mu_{A\cup B}^l(x), \mu_{A\cup B}^r(x)\right]$$
$$\mu_{A\cup B}^l(x) = \max\left\{\mu_A^l(x), \mu_B^l(x)\right\} \quad (4.7)$$
$$\mu_{A\cup B}^r(x) = \max\{\mu_A^r(x), \mu_B^r(x)\}$$

$$A \cap B = \left\{\left(x,\ \mu_{A\cap B}(x)\right) \mid x \in X\right\}$$
$$\mu_{A\cap B}(x) = \left[\mu_{A\cap B}^l(x), \mu_{A\cap B}^r(x)\right]$$
$$\mu_{A\cap B}^l(x) = \min\left\{\mu_A^l(x), \mu_B^l(x)\right\} \quad (4.8)$$
$$\mu_{A\cap B}^r(x) = \min\{\mu_A^r(x), \mu_B^r(x)\}$$

$$\bar{A} = \left\{\left(x,\ \mu_{\bar{A}}(x)\right) \mid x \in X\right\}$$
$$\mu_{\bar{A}}(x) = \left[\mu_{\bar{A}}^l(x), \mu_{\bar{A}}^r(x)\right]$$
$$\mu_{\bar{A}}^l(x) = 1 - \mu_{\bar{A}}^r(x) \quad (4.9)$$
$$\mu_{\bar{A}}^r(x) = 1 - \mu_{\bar{A}}^l(x)$$

Aralık değerli bulanık kümelerde, üyelik derecelerini tayin etme sırasındaki belirsizlikleri ifade etmek için, aralık değerleri üyelik derecesi olarak kullanılır.

Aralığın geniş olması üyelik derecesindeki belirsizliğin fazla olduğunu gösterir (Lee, 2004).

4.2.2 Sezgisel bulanık kümeler

Sezgisel bulanık küme teorisi 1986 yılında Atanassov tarafından geliştirilen bir bulanık küme teorisi uzantısıdır (Lee, 2004).

Sezgisel bulanık değerler; üyelik derecesi, üye olmama derecesi ve tereddüt derecesi gibi daha kapsamlı bir değerlendirmeyi yansıttığından, sezgisel bulanık kümelerin belirsizlik konusundaki faydasının bulanık kümelerden daha iyi olduğu açıktır (Zhang ve diğ, 2011). Sıradan bulanık kümelerin uzantısı olan sezgisel bulanık kümeler, karar vermede yer alan belirsizlik ve belirsizliği geliştirmek için esnek bir model sağlarlar (Öztayşi ve diğ, 2015).

X evrenindeki sezgisel A bulanık kümesi, aşağıdaki üyelik fonksiyonuyla tanımlanır:

$$A = \{(x,\ \mu_A(x),\ V_A(x)) \mid x \in X\},$$
$$\mu_A(x) : X \rightarrow [0,1], \qquad V_A(x) : X \rightarrow [0,1] \tag{4.10}$$

$\mu_A(x)$ üyelik derecesini ifade ederken, $V_A(x)$ üye olmama derecesini gösterir ve

$$x \in X \text{ için } 0 \leq \mu_A(x) + V_A(x) \leq 1 \tag{4.11}$$

$\pi_A = 1 - (\mu_A(x) + V_A(x))$ değeri tereddüt kısmı ya da sezgisellik ölçütü olarak ifade edilir. Yani, sezgisel kümeler üyelik değerlerini tayin ederkenki belirsizliği ifade etmek için kullanılan bir gösterimdir (Lee, 2004).

A ve B'nin iki sezgisel bulanık küme olduğu varsayılırsa, sezgisel bulanık kümelerin temel işlemleri aşağıdaki gibi tanımlanmıştır:

$$A \cup B = \{(x,\ \mu_{A\cup B}(x),\ V_{A\cup B}(x)) \mid x \in X\}$$
$$\mu_{A\cup B}(x) = \max\{\mu_A(x), \mu_B(x)\} \tag{4.12}$$
$$V_{A\cup B}(x) = \min\{V_A(x), V_B(x)\}$$

$$A \cap B = \{(x,\ \mu_{A\cap B}(x),\ V_{A\cap B}(x)) \mid x \in X\}$$

$$\mu_{A\cap B}(x) = \min\{\mu_A(x), \mu_B(x)\} \tag{4.13}$$

$$V_{A\cap B}(x) = \max\{V_A(x), V_B(x)\}$$

$$\bar{A} = \{(x,\ \mu_{\bar{A}}(x),\ V_{\bar{A}}(x)) \mid x \in X\}$$

$$\mu_{\bar{A}}(x) = V_A(x) \tag{4.14}$$

$$V_{\bar{A}}(x) = \mu_A(x)$$

Lee (2004), aralık değerli bulanık kümelerle sezgisel bulanık kümeleri karşılaştırdığında, ikisinin de aynı temel operasyonlara sahip olduğu, dolayısıyla belirsizliği ifade etmede eşit güce sahip olduğunu görmüştür.

4.2.3 Bipolar değerli bulanık kümeler

Bipolar değerli bulanık kümeler, üyelik derece aralığının [0,1]'den [-1,1]'e genişletildiği bir bulanık küme uzantısıdır (Lee, 2004). Bipolar değerli bulanık kümelerde 0 üyelik derecesi, elemanın ilgili özellikle alakasız olduğunun göstergesidir. (0,1] aralığı, elemanların bir şekilde özelliği karşıladığını, [-1, 0) aralığı ise elemanların bir şekilde karşıt özelliği karşıladığını belirtir. Bipolar değerli bulanık kümelerde iki farklı gösterim biçimi kullanılır; doğal (kanonik) ve indirgenmiş. Doğal gösterim biçiminde üyelik dereceleri, pozitif üyelik ve negatif üyelik değerleri çiftiyle ifade edilir. Dolayısıyla üyelik dereceleri iki kısma bölünür: [0,1] aralığındaki pozitif kısım ve [-1, 0] aralığındaki negatif kısım. İndirgenmiş gösterimde ise üyelik derecesi [-1, 1] aralığındaki bir değerle gösterilir (Lee, 2004).

X evrenindeki bipolar değerli A bulanık kümesi, doğal gösterimde aşağıdaki gibi tanımlanır:

$$A = \left\{\left(x,\ \left(\mu_A^P(x),\ \mu_A^N(x)\right)\right) \mid x \in X\right\},$$

$$\mu_A^P(x) : X \to [0,1], \qquad \mu_A^N(x) : X \to [-1,0] \tag{4.15}$$

$\mu_A^P(x)$, x elemanının A bipolar bulanık kümesindeki bir özelliği karşılama derecesini gösterirken; $\mu_A^N(x)$, x elemanının A bipolar bulanık kümesindeki karşıt özelliği karşılama derecesini gösterir.

$\mu_A^P(x) \neq 0$ ve $\mu_A^N(x) = 0$ ise, x elemanı A kümesi için sadece pozitif karşılama özelliğine sahiptir. $\mu_A^P(x) = 0$ ve $\mu_A^N(x) \neq 0$ olduğu durumda ise, x A'nın bir özelliğini karşılamaz; ancak A'nın karşıt bir özelliğini bir şekilde karşılıyor demektir (Lee, 2004).

Doğal gösterimde, özelliğe üyelik fonksiyonunun, karşıt özelliğe üyelikle kesiştiği durumlarda, x elemanı için $\mu_A^P(x) \neq 0$ ve $\mu_A^N(x) \neq 0$ olması mümkündür.

X evrenindeki bipolar değerli A bulanık kümesi, indirgenmiş gösterimde aşağıdaki gibi tanımlanır:

$$A = \{(x,\ \mu_A^R) \mid x \in X\}$$
$$\mu_A^R : X \rightarrow [-1,1] \tag{4.16}$$

A ve B 'nin bipolar değerli bulanık kümeler olduğunu varsayarsak, x elemanının üyelik derecesi doğal gösterimle aşağıdaki gibi ifade edilir:

$$A = \left\{\left(x,\ \left(\mu_A^P(x),\ \mu_A^N(x)\right)\right) \middle|\ x \in X\right\}$$
$$B = \left\{\left(x,\ \left(\mu_B^P(x),\ \mu_B^N(x)\right)\right) \middle|\ x \in X\right\} \tag{4.17}$$

Bipolar değerli bulanık kümelerin temel işlemleri aşağıdaki gibi tanımlanmıştır:

$$A \cup B = \left\{\left(x,\ \mu_{A\cup B}(x)\right) \middle|\ x \in X\right\}$$
$$\mu_{A\cup B}(x) = \left(\mu_{A\cup B}^P(x), \mu_{A\cup B}^N(x)\right)$$
$$\mu_{A\cup B}^P(x) = \max\{\mu_A^P(x), \mu_B^P(x)\} \tag{4.18}$$
$$\mu_{A\cup B}^N(x) = \min\{\mu_A^N(x), \mu_B^N(x)\}$$

$$A \cap B = \left\{\left(x,\ \mu_{A\cap B}(x)\right) \middle|\ x \in X\right\}$$
$$\mu_{A\cap B}(x) = \left(\mu_{A\cap B}^P(x), \mu_{A\cap B}^N(x)\right) \tag{4.19}$$

$$\mu^P_{A\cap B}(x) = \min\{\mu^P_A(x), \mu^P_B(x)\}$$

$$\mu^N_{A\cap B}(x) = \max\{\mu^N_A(x), \mu^N_B(x)\}$$

$$\bar{A} = \{(x,\ \mu_{\bar{A}}(x)) \mid x \in X\}$$

$$\mu_{\bar{A}}(x) = \left(\mu^P_{\bar{A}}(x), \mu^N_{\bar{A}}(x)\right) \tag{4.20}$$

$$\mu^P_{\bar{A}}(x) = 1 - \mu^P_A(x)$$

$$\mu^N_{\bar{A}}(x) = -1 - \mu^N_A(x)$$

Lee (2004), sezgisel bulanık kümelerle bipolar değerli bulanık kümeleri karşılaştırdığında, iki yaklaşım benzer gözükse de kullanım amaçlarının farklı olduğu sonucuna varmıştır. Bipolar değerli bulanık kümelerde negatif üyelik derecesi $\mu^N_A(x)$, x elemanının, karşıt A özelliğini ne kadar karşıladığını gösterirken; sezgisel bulanık kümelerde $V_A(x)$, x elemanının, A özelliğini karşılamama derecesini gösterir.

Genellikle "karşıt özellik", bir "özelliği karşılamama" ile aynı olmadığından bipolar değerli bulanık kümelerle sezgisel bulanık kümeler, bulanık kümelerin farklı uzantılarıdır. Sezgisel bulanık kümeler üyelik derecelerini tayin ederken belirsizlik olduğu durumlarda kullanılmaya elverişli iken, bipolar değerli bulanık kümeler alakasız ve zıt elemanların ayrıştırılmasının gerektiği durumlarda kullanılmalıdır (Lee, 2004).

4.2.4 Aralık-değerli sezgisel bulanık kümeler (ADSBK)

Günlük hayattaki birçok durumda, mevcut bilgilerin yetersizliği nedeniyle bir elemanın belirli bir kümeye üyelik ve üye olmama dereceleri için kesin değerler belirlemek kolay olmayabilir. Bu nedenle Atanassov ve Gargov (1989), sezgisel bulanık küme teorisini geliştirerek, değerleri gerçek sayılar yerine aralıklar olan ve, üyelik ve üye olmama fonksiyonları ile karakterize edilen, ADSBK kavramını ortaya çıkarmışlardır (Li, 2011). ADSBK'ler, bulanık mantığın üyelik ve üye olmama dereceleri kavramının kesin sayısal değerlerle ifade edilemediği karmaşık karar verme problemlerinin çözümü için uygun gözükmektedir (Li, 2011).

Aralık değerli sezgisel A bulanık kümesi, aşağıdaki üyelik fonksiyonuyla tanımlanır (Lee, 2004; Ahn ve diğ, 2011):

$$A = \{(x,\ \mu_A(x),\ V_A(x)) \mid x \epsilon X\}, \qquad \mu_A : X \rightarrow [I]\ ve\ V_A : X \rightarrow [I] \tag{4.21}$$

$\mu_A(x)$ üyelik derecesini ifade ederken, $V_A(x)$ üye olmama derecesini gösterir ve

$$x \epsilon X\ için\ 0 \leq \mu_A(x) + V_A(x) \leq 1 \tag{4.22}$$

$\mu_A(x) = [\mu_A^l(x), \mu_A^r(x)]$ ve $V_A(x) = [V_A^l(x), V_A^r(x)]$ olduğundan,

$$A = ([\mu_A^l(x), \mu_A^r(x)], [V_A^l(x), V_A^r(x)]) \tag{4.23}$$

4.3 Bulanık AHS

Analitik Hiyerarşi Süreci (AHS), ilgili olabilecek özelliklerin göreceli önemlerine göre toplamsal olarak ağırlıklandırılarak değerlendirildiği çok kriterli bir analitik süreçtir. AHS aracılığıyla kriterlerin önemi, maddi olmayan özelliklerin veya özelliklerin kategorilerinin hiyerarşik bir yapıda ikili olarak karşılaştırıldığı bir işlemle elde edilir. (Chou ve diğ. 2014). AHS vasıtasıyla, ikili karşılaştırmalar şeklinde uzman görüşleri alınır ve karşılaştırma matrislerinden oran ölçekleri çıkarılır. Karşılaştırma matrisleri, karar vericilerin farklı alternatifler arasında değerlendirme kriterlerini ve bu kritelerin ağırlıkları gözetilerek yapılan tercihleri belirtir. Kriterlere verilen puanlar bazında her bir alternatif için hesaplanan normalleştirilmiş ağırlıklı toplam, karar vericini en iyi alternatifi seçmesine yardımcı olur (Ahmed ve Kılıç, 2018).

Karmaşık herhangi bir problem, AHS vasıtasıyla alt problemlere bölünebilir. Her hiyerarşik seviye, her alt problemin kriter ya da özellik setini temsil eder (Chou ve diğ. 2014).

AHS genelde kesin bilgiye sahip olunan durumlarda karar vermek için kullanılır; çünkü AHS, dengesiz bir yargılama ölçeği kullanır ve doğal dildeki insan yargısının bir sayıyla eşleştirilmesi sırasında ortaya çıkan belirsizliği gözetmez. AHS sıralama yöntemi, karar vericilerin algısına ve öznel değerlendirmelerine dayalı olduğundan ortaya çıkan sonuçlar kesinlikten oldukça uzaktır (Chou ve diğ. 2014). Geleneksel

AHS yöntemleri, çok kriterli karar verme problemlerinde sayısal ve niteliksel kriterleri değerlendirmede tutarlı olmasına rağmen, karar vericinin yargılarındaki bulanıklığı ve belirsizliği de kesin gibi değerlendirmeye katmaktadır (Sheu, 2004).

Literatüre göre ikili karşılaştırmaları tutmak için genelde iki farklı ölçek kullanılmıştır, (1-9 arasındaki) kesin sayılara dayalı ölçek ve bulanık sayılara dayalı ölçek (Ahmed ve Kılıç, 2018). Saaty tarafından önerilen orijinal yöntemde 1-9 arasındaki kesin sayı ölçeği kullanılmıştır ve karar vericiler tarafından "zayıf", "normal", "güçlü" vb.şekilde doğal dilde ifade edilen değerlendirmelerin tek ve keskin bir karşılığı vardır (Ahmed ve Kılıç, 2018).

Bahsedilen bu problemlere çözüm bulmak için AHS ile bulanık mantık teorileri bir araya getirilerek, karar vericinin öznel yargılarıyla kriter ağırlıklarını belirlemeye ilişkin çalışmalar yapılmıştır ve bu çalışmalar Bulanık AHS olarak adlandırılmıştır (Chou ve diğ. 2014; Memik, 2017). Çok Kriterli Karar Verme (ÇKKV) süreçlerinde AHS ve bulanık küme teorisi, bulanık sayıların insan yargısını daha gerçekçi bir şekilde yansıtması nedeniyle yaygın olarak kullanılmıştır (Ahmed ve Kılıç, 2018).

Bulanık AHS yönteminde klasik AHS'den farklı olarak ikili karşılaştırma oranlarının belirlenmesinde kesin sayılar yerine değer aralıkları kullanılmıştır. Karar vericilerin bir alternatifle ilgili yargılarını kesin bir sayı ile ifade etmeleri yerine, günlük hayata daha uygun olan sözel değerlendirmelerle yapmaları ve bu değerlendirmelerin bulanık sayılarla temsil edilmesi daha sağlıklıdır (Gu ve Zhu, 2006). Bulanık AHS'de ağırlıklar, bulanık karşılaştırma matrisleriyle hesaplanır ve her bir alternatifin her bir kriter için aldığı puan bu ağırlıklar doğrultusunda hesaplanarak, alternatifler arasında sıralama yapılır (Ahmed ve Kılıç, 2018).

4.4 Tercih İlişkileri

Karar verme sürecinde karar verici genelde her bir alternatif çifti için bir tercih yapmak zorunda kalır ve böylelikle bir tercih ilişkisi oluşur. $X = \{x_1, x_2, \dots, x_n\}$ alternatiflerin ayrık kümesi olduğunda, ilişki aşağıdaki gibi tanımlanır (Xu, 2007):

$$\mu_P = X \times X \rightarrow D \tag{4.24}$$

Burada D, tercih derecelerinin tanım kümesi iken; P, X kümesindeki tercih ilişkisidir.

Tercih ilişkileriyle karar verme problemleriyle ilgili birçok akademik çalışma yapılmıştır. Tercih ilişkileri temel olarak aşağıdaki 3 kategoride sınıflandırılabilir (Xu, 2007):

4.4.1 Çarpımsal tercih ilişkileri

X kümesindeki çarpımsal tercih ilişkisi ters matris P ile aşağıdaki gibi tanımlanır:

$$P = (p_{ij})_{n\times n} \subset X \times X \quad ve$$
$$p_{ij} > 0, \qquad p_{ij}\,.\,p_{ji} = 1, \qquad p_{ii} = 1, \qquad \mathrm{i,j} = 1,2,\dots,\mathrm{n} \tag{4.25}$$

Burada p_{ij}, alternatif x_i'nin alternatif x_j'ye tercih oranını gösterir.

Dolayısıyla, $p_{ij} = 1$ olması, alternatif x_i ile alternatif x_j arasında bir fark olmadığını, $p_{ij} > 1$ olması, alternatif x_i'nin alternatif x_j'ye tercih edildiğini belirtir.

4.4.2 Bulanık tercih ilişkileri

X kümesindeki bulanık tercih ilişkisi tümleyen matris R ile aşağıdaki gibi tanımlanır:

$$R = (r_{ij})_{n\times n} \subset X \times X \quad ve$$
$$r_{ij} \geq 0, \qquad r_{ij} + r_{ji} = 1, \qquad r_{ii} = 0.5, \qquad \mathrm{i,j} = 1,2,\dots,\mathrm{n} \tag{4.26}$$

Burada r_{ij}, alternatif x_i'nin alternatif x_j'ye tercih oranını gösterir.

Dolayısıyla, $r_{ij} = 0.5$ olması, alternatif x_i ile alternatif x_j arasında bir fark olmadığını, $p_{ij} > 0.5$ olması, alternatif x_i'nin alternatif x_j'ye tercih edildiğini belirtir.

4.4.3 Dilsel tercih ilişkileri

s_i'nin dilsel bir değişkeni temsil ettiği durumda, sonlu ve ayrık dilsel etiket kümesi $S = \{s_i |\, i = -t, \dots, t\}$ 'nin aşağıdaki 2 özelliği sağladığını düşünelim:

i > j ise $s_i > s_j$

Olumsuzlama operatörü neg $(s_t) = s_{-t}$

S kümesinin eleman sayısı, karar vericiyi gereksiz bir kesinliğe zorlamayacak kadar az, her bir alternatifin performansını ayırt edebilecek kadar çok olmalıdır.

Ayrık etiket kümesi S, devamlı etiket kümesi $\bar{S} = \{s_\alpha | \ \alpha \in [-q, q]\}$ 'ye genişletildiğinde, X kümesindeki dilsel tercih ilişkisi, dilsel karar matrisi L ile aşağıdaki gibi tanımlanır:

$$L = \left(l_{ij}\right)_{n\times n} \subset X \times X \quad ve$$

$$l_{ij} \in \bar{S}, \qquad l_{ij} \oplus l_{ji} = s_0, \qquad l_{ii} = s_0, \qquad \mathrm{i,j} = 1,2,\dots,\mathrm{n} \tag{4.27}$$

Burada l_{ij}, alternatif x_i'nin alternatif x_j'ye tercih oranını gösterir.

Dolayısıyla, $l_{ij} = s_0$ olması, alternatif x_i ile alternatif x_j arasında bir fark olmadığını, $\mathrm{l}_{ij} > s_0$ olması, alternatif x_i'nin alternatif x_j'ye tercih edildiğini belirtir.

4.5 Sezgisel Bulanık Tercih İlişkileri

Günlük hayattaki birçok durumda, karar verici, alternatifler arasında tercihini bir dereceye kadar belirlese de çoğu durumda yaptığı tercihten kesin olarak emin olamaz. Bu nedenle tercih derecesini gerçek sayısal değerler ya da dilsel değişkenlerle belirtmek yerine sezgisel bulanık değerlerle ifade etmesi daha uygun olur. Bu nedenle bulanık tercih ilişkisi, Szmidt ve Kacprzyk tarafından sezgisel bulanık tercih ilişkisi olarak genişletilmiştir (Xu, 2007).

X kümesindeki sezgisel tercih ilişkisi, B matrisi ile aşağıdaki gibi tanımlanır:

$$B = \left(b_{ij}\right)_{n\times n} \subset X \times X \quad ve$$

$$b_{ij} = < \left(x_i, x_j\right), \mu\left(x_i, x_j\right), \upsilon\left(x_i, x_j\right) >, \qquad \mathrm{i,j} = 1,2,\dots,\mathrm{n} \tag{4.28}$$

Burada $b_{ij} = \left(\mu_{ij}, \nu_{ij}\right)$ sezgisel bulanık bir değerse, μ_{ij}, x_i alternatifinin x_j alternatifine tercih derecesini gösterirken; ν_{ij}, x_i alternatifinin x_j alternatifine tercih edilmeme derecesini gösterir ve $\pi_{ij} = 1 - \mu_{ij} - \nu_{ij}$; x_i alternatifinin x_j alternatifine tercih edilmesindeki belirsizlik derecesidir (Xu, 2007). Ayrıca μ_{ij} ve ν_{ij} aşağıdaki özelliklere sahiptir:

$$0 \leq \mu_{ij} + \nu_{ij} \leq 1, \qquad \mu_{ji} = \nu_{ij},\ \nu_{ji} = \mu_{ij,} \quad \mu_{ii} = \nu_{ii} = 0.5 \tag{4.29}$$

Xu (2007) sezgisel tercih ilişkilerini birleştirebilmek için aşağıdaki bağıntı ve işlemleri tanıtmıştır:

Tüm $i, j, k, l = 1, 2, \dots, n$ *için*, $b_{ij} = (\mu_{ij}, \nu_{ij})$ *ve* $b_{kl} = (\mu_{kl}, \nu_{kl})$ ise:

(1) $\overline{b_{ij}} = (\nu_{ij}, \mu_{ij})$

(2) $b_{ij} + b_{kl} = (\mu_{ij} + \mu_{kl} - \mu_{ij}.\mu_{kl}, \nu_{ij}.\nu_{kl})$

(3) $b_{ij}.b_{kl} = (\mu_{ij}.\mu_{kl},\ \nu_{ij} + \nu_{kl} - \nu_{ij}.\nu_{kl})$

(4) $\lambda b_{ij} = \left(1 - (1 - \mu_{ij})^{\lambda}, {\nu_{ij}}^{\lambda}\right),\ \lambda > 0$

(5) ${b_{ij}}^{\lambda} = \left({\mu_{ij}}^{\lambda}, 1 - (1 - \nu_{ij})^{\lambda}\right),\ \lambda > 0$

(6) $b_{ij} + b_{kl} = b_{kl} + b_{ij}$

(7) $b_{ij}.b_{kl} = b_{kl}.b_{ij}$

(8) $\lambda(b_{ij} + b_{kl}) = \lambda b_{ij} + \lambda b_{kl},\ \lambda > 0$

(9) $(b_{ij}.b_{kl})^{\lambda} = {b_{ij}}^{\lambda}.{b_{kl}}^{\lambda},\ \lambda > 0$

(10) $\lambda_1 b_{ij} + \lambda_2 b_{ij} = (\lambda_1 + \lambda_2) b_{ij}, \lambda_1, \lambda_2 > 0$

(11) ${b_{ij}}^{\lambda_1}.{b_{ij}}^{\lambda_2} = {b_{ij}}^{\lambda_1 + \lambda_2}$

Ayrıca B sezgisel tercih ilişkisi, çarpımsal geçişliliği sağladığında:

Tüm $i, j, k = 1, 2, \dots, n$ *için*, $b_{ij} = b_{ik}.b_{kj}$ (4.30)

Bu durumda B'ye tutarlı B sezgisel tercih ilişkisi denir (Xu, 2007).

Xu (2007) iki sezgisel bulanık değeri karşılaştırabilmek için skor (Δ) ve kesinlik (H) fonksiyonlarını tanımlamıştır. $b_{ij} = (\mu_{ij}, \nu_{ij})$ için:

$$\Delta(b_{ij}) = \mu_{ij} - \nu_{ij}\ , \Delta(b_{ij}) \in [-1,1] \tag{4.31}$$

$\Delta(b_{ij})$ değeri arttıkça sezgisel bulanık sayı b_{ij} 'nin değeri artar.

$$H\left(b_{ij}\right) = \mu_{ij} + \nu_{ij}\ , H\left(b_{ij}\right) \in [0,1] \qquad (4.32)$$

$H\left(b_{ij}\right)$ değeri arttıkça sezgisel bulanık sayı b_{ij} 'nin kesinlik derecesi artar.

Xu (2007), daha sonra, skor (Δ) ve kesinlik (H) fonksiyonlarından faydalanarak herhangi iki sezgisel bulanık değeri sıralayabilmek için bir sıralama ilişkisi tanımlar:

Eğer $\Delta\left(b_{ij}\right) < \Delta\left(b_{kl}\right)$ ise $b_{ij} < b_{kl}$,

Eğer $\Delta\left(b_{ij}\right) = \Delta\left(b_{kl}\right)$ ise ve

$H\left(b_{ij}\right) = H\left(b_{kl}\right)$ ise $b_{ij} = b_{kl}$,

$H\left(b_{ij}\right) < H\left(b_{kl}\right)$ ise $b_{ij} < b_{kl}$

Xu (2007), son olarak, ağırlıklara göre birleştirilmiş sezgisel bulanık değer $\overline{b_{ij}}$ 'yi aşağıdaki gibi tanımlar:

$w = (w_1, w_2, \dots, w_m)^T$, ${b_{ij}}^{(1)}, {b_{ij}}^{(2)}, \dots, {b_{ij}}^{(m)}$ 'nin ağırlık vektörü olduğunda ve

$w_k > 0$, k =0, 1, ..., k, $\sum_{k=1}^{m} w_k = 1$ iken,

sezgisel bulanık ağırlıklandırılmış aritmek ortalama işlemiyle:

$$\overline{b_{ij}} = \sum_{k=1}^{m} w_k {b_{ij}}^{(k)} \ , \ \text{i,j= 1,2,...,n} \qquad (4.33)$$

sezgisel bulanık ağırlıklandırılmış geometrik ortalama işlemiyle:

$$\overline{b_{ij}} = \prod_{k=1}^{m} {b_{ij}}^{(k)^{w_k}} \ , \ \text{i,j= 1,2,...,n} \qquad (4.34)$$

Eğer tüm sezgisel bulanık değerlerin ağırlıkları eşitse, yani $w = (1/m, 1/\text{m}, \dots, 1/\text{m})^T$ ise:

sezgisel bulanık ağırlıklandırılmış aritmek ortalama işlemiyle:

$$\overline{b_{ij}} = \frac{1}{m}\sum_{k=1}^{m} {b_{ij}}^{(k)} \ , \ \text{i,j= 1,2,...,n} \qquad (4.35)$$

sezgisel bulanık ağırlıklandırılmış geometrik ortalama işlemiyle:

$$\overline{b_{ij}} = \left(\prod_{k=1}^{m} {b_{ij}}^{(k)}\right)^{1/m} \text{ , i,j= 1,2,...,n} \quad (4.36)$$

4.5.1 Tamamlanmamış sezgisel tercih ilişkileri

Xu (2007), karar vericinin zaman baskısı, bilgi eksikliği ya problemle ilgili kısıtlı tecrübesi olmasından dolayı, sezgisel tercih ilişkilerini oluştururken tamamlanmamış yargılarda bulunabileceği düşüncesiyle, karar verme problemini tamamlanmamış sezgisel tercih ilişkileriyle araştırmış ve $B = \left(b_{ij}\right)_{n\times n}$ matrisinin bazı elemanları karar verici tarafından belirlenmediyse, B'yi tamamlanmamış sezgisel tercih ilişkisi olarak tanımlamıştır.

Çarpımsal geçişliliği sağlayan tutarlı bir B ilişkisi için, eğer (i,j) ∩ (k,l) ≠ ∅ ise b_{ij} ve b_{kl} komşu elemanlar olur ve eğer bilinmeyen bir b_{ij} elemanı için bilinen iki komşu b_{ik} ve b_{kj} elemanı mevcutsa b_{ij} değerine aşağıdaki formülle ulaşılabilir:

$$b_{ij} = b_{ik}.b_{kj} \quad (4.37)$$

Tamamlanmamış sezgisel tercih ilişkisi B'nin diyagonal elemanları haricinde, her bir sütun ya da satırında en az bir bilinen eleman varsa, bilinmeyen elemanlar yukarıdaki formülle hesaplanabilir ve bu B ilişkisi "kabul edilebilir" olarak tanımlanır.

Eğer bir B sezgisel tercih ilişkisi, hem karar verici tarafından direkt olarak belirlenmiş tercihleri hem de bilinmeyen elemanların bilinen elemanlar vasıtasıyla hesaplandığı tercihleri içeriyorsa, bu ilişkiye geliştirilmiş sezgisel tercih ilişkisi denir ve $\bar{B}$ ile gösterilir (Xu, 2007).

4.5.2 Tamamlanmamış aralık değerli sezgisel tercih ilişkileri (TADSTİ)

Xu ve Che (2007) tercih ilişkisini oluştururken karar vericinin bilgisini kesin sayılarla ifade etmesindeki zorluğu göz önünde bulundurarak, tamamlanmamış sezgisel tercih ilişkilerini, aralık değerlerini sezgisel bulanık kümeleri kullanarak geliştirmiş ve aşağıdaki tanımı yapmışlardır:

$$\tilde{Q} = \left(\tilde{q}_{ij}\right)_{n\times n},\ \tilde{q}_{ij} = \left(\tilde{\mu}_{ij}, \tilde{v}_{ij}\right) ve \quad (4.38)$$

$$\tilde{\mu}_{ij} = \left[\tilde{\mu}_{ij}{}^{L}, \tilde{\mu}_{ij}{}^{U}\right] \subset [0,1], \qquad \tilde{\nu}_{ij} = \left[\tilde{\nu}_{ij}{}^{L}, \tilde{\nu}_{ij}{}^{U}\right] \subset [0,1],$$
$$\tilde{\mu}_{ji} = \tilde{\nu}_{ij},\ \nu_{ji} = \tilde{\mu}_{ij},$$
$$\tilde{\mu}_{ij}{}^{U} + \tilde{\nu}_{ij}{}^{U} \leq 1,\ \ \mathrm{i,j} = 1,2,\dots,\mathrm{n}$$

Burada $\tilde{Q}$, TADSTB ve $\tilde{q}_{ij}$, bir Aralık Değerli Sezgisel Bulanık Sayı (ADSBS)'dır. $\tilde{\mu}_{ij}$, karar vericinin A_i alternatifini A_j alternatifine tercih etme derece aralığını gösterirken; $\tilde{\nu}_{ij}$, karar vericinin A_j alternatifini A_i alternatifine tercih etme derece aralığını gösterir.

Bilinmeyen elemanların hesaplanması için kullanılan (4.37)'deki denklem, Xu ve Cai tarafından 2009'ta aşağıdaki denklemlerle ifade adilmiştir (Öztayşi ve diğ, 2015'te atıfta bulunulduğu gibi):

$$\tilde{q}_{ij} = \frac{\tilde{q}_{ik} \oplus \tilde{q}_{kj}}{2} \tag{4.39}$$

$$\tilde{q}_{ij} = \sqrt{\left(\tilde{q}_{ik} \otimes \tilde{q}_{kj}\right)} \tag{4.40}$$

Xu ve Chen (2007), aralık değerli sezgisel tercih ilişkilerinde birleştirme yapabilmek için Aralık-Değerli Sezgisel Bulanık Ağırlıklandırılmış Birleştirme (ASBAB) işlemini aşağıdaki gibi tanımlamıştır. Bu işlem sonucunda her bir alternatif için birleştirilmiş bir ADSBS elde edilir.

$$\begin{aligned} ASBAB_w\left(\tilde{q}_{ij}{}^{(1)}, \tilde{q}_{ij}{}^{(2)}, \dots, \ \ \tilde{q}_{ij}{}^{(n)}\right) \\ = \left(\left[1 - \prod_{k=1}^{n}\left(1 - \tilde{\mu}_k{}^{L}\right)^{w_k},\ \ 1 - \prod_{k=1}^{n}\left(1 - \tilde{\mu}_k{}^{U}\right)^{w_k}\right],\right. \\ \left.\left[\prod_{k=1}^{n} \tilde{\nu}_j{}^{L^{w_k}}, \tilde{\nu}_j{}^{U^{w_k}}\right]\right) \end{aligned} \tag{4.41}$$

Burada $\mathrm{w} = (w_1, w_2, \dots, w_n)^T$, $\tilde{q}_{ij}{}^{(1)}, \tilde{q}_{ij}{}^{(2)}, \dots, \tilde{q}_{ij}{}^{(n)}$ 'nin ağırlık vektörü ve $w_k > 0$, k =0, 1, ..., n, $\sum_{k=1}^{n} w_k = 1$ 'dir.

Xu ve Chen (2007) Sıralanmış Ağırlıklandırılmış Birleştirme (SAB) operatörü w_G'yi şöyle tanımlamışlardır:

$$w_{G_j} = \frac{e^{-[(j-\mu_G)^2/2\sigma_G{}^2]}}{\sum_{i=1}^{n} e^{-[(i-\mu_G)^2/2\sigma_G{}^2]}} \quad (4.42)$$

Burada, orta değer $\mu_G = \frac{1+n}{2}$ ve standart sapma $\sigma_G = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(i-\mu_G)^2}$ 'dir.

Alternatiflerin birleştirilmiş değerleri arasında bir sıralama yapabilmek için skor ve kesinlik fonksiyonları kullanılır. Xu ve Chen (2007), Xu'nun 2007'de tanımladığı skor ve kesinlik fonksiyonlarını aralık değerli sezgisel bulanık sayılar için aşağıdaki gibi tanımlamıştır:

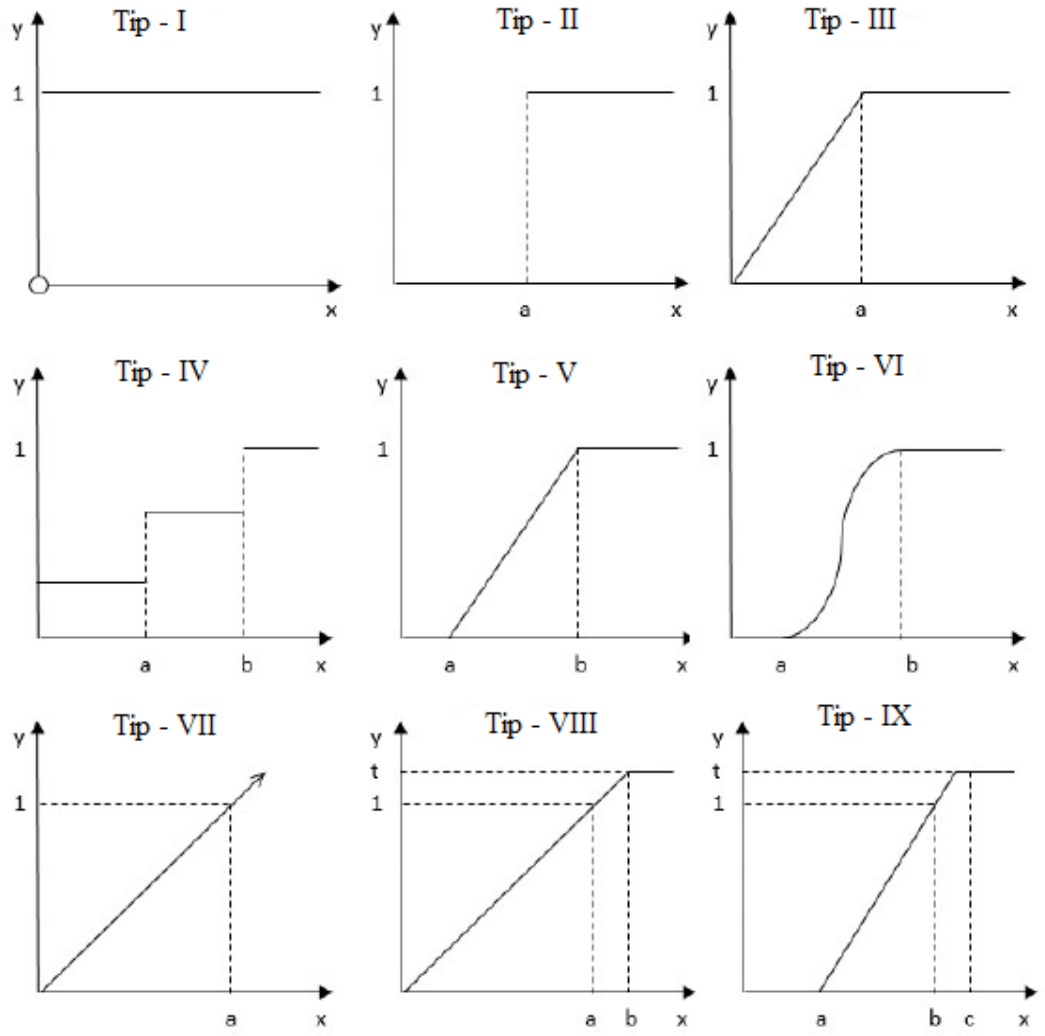
$$S(\tilde{q}) = \frac{\tilde{\mu}^L - \tilde{v}^L + \tilde{\mu}^U - \tilde{v}^U}{2} \quad (4.43)$$

$$H(\tilde{q}_{ij}) = \frac{\tilde{\mu}^L + \tilde{v}^L + \tilde{\mu}^U + \tilde{v}^U}{2} \quad (4.44)$$

4.6 Performans Gösterge Fonksiyonları

Performans değerlendirmesi sırasında değerlendirilen kişilerin sayısı arttıkça ve yeterlilikleri değişkenlik gösterdikçe, zaman faktörü de göz önünde bulundurulursa ikili karşılaştırmalarla performans değerlendirmesi yapmak oldukça zor ve adaletsiz olmaktadır (Öztayşi ve diğ, 2016). Bu duruma çözüm bulmak için Öztayşi ve diğ, (2015), Brans ve Vincke tarafından 1985'te PROMETHEE (Değerlendirmenin Zenginleştirilmesi için Tercih Sırası Düzenleme Metodu) metodunda kullanılan tercih fonksiyonlarını temel alarak, performans ölçme alanında kullanılmak üzere performans gösterge fonksiyonlarını tanımlamışlardır. Önerdikleri performans gösterge fonksiyonları Şekil 4.4'teki gibidir.

Performans gösterge fonksiyonları, gözlemlenen performans değeri ile boyutsuz performans skoru arasındaki ilişkiyi gösterir. Grafiksel gösterimde, x ekseni gerçek gösterge değerlerini simgelerken, y ekseni performans skorunu simgeler. Fonksiyonların açıklamaları aşağıdaki gibidir (Öztayşi ve diğ, 2016).



Şekil 4.4 : Performans gösterge fonksiyonları.

- Tip – I (sıradan kriter): 0'dan büyük herhangi bir gösterge değeri çin performans skoru 1'dir.
- Tip – II (benzer kriter): a eşik değerinden büyük herhangi bir gösterge değeri için performans skoru 1'dir.
- Tip – III (v-şeklinde kriter): a eşik değerden büyük herhangi bir gösterge değeri için performans skoru 1'dir ve x_i gözlemlenen gösterge değeri iken, a eşik değerinin altındaki gösterge değerleri için performans skoru $\frac{x_i}{a}$ formülüyle hesaplanır.

- Tip – IV (seviye kriteri): Gözlemlenen gösterge değerleri için farklı aralıklar tanımlanır ve her bir aralık için farklı performans skorları atanır.
- Tip – V (sözde kriter): Hesaplama Tip – III'e benzerdir; ancak a eşik değerinin altındaki gösterge değerleri için performans skoru 0'dır.
- Tip – VI (Gauss kriteri): Hesaplama Tip – V'e benzerdir; ancak a ve b parametreleri arasındaki gösterge değerleri için performans skoru Gauss fonksiyonuyla hesaplanır.
- Tip – VII (limitsiz ibre kriteri): Gözlemlenen gösterge değeri a iken performans skoru 1'dir. Diğer x_i gözlemlenen gösterge değerleri için, performans skoru $\frac{x_i}{a}$ formülüyle hesaplanır. Bu fonksiyonun Tip III'ten farkı, performans skorunun 1'den büyük olabilmesidir.
- Tip – VIII (genişletilmiş v-şeklinde kriter): Hesaplama Tip III'e benzerdir; ancak maksimum performans skor limiti t, 1'den büyüktür.
- Tip – IX (genişletilmiş sözde kriter): Hesaplama Tip V'e benzerdir; ancak maksimum performans skor limiti t, 1'den büyüktür.

4.7 Uygulamada Kullanılan Metodun Adımları

Uygulama sırasında çalışanların performansını değerlendirmek için kullanılan kriterlerin ağırlıkları, Tamamlanmamış Aralık Değerli Sezgisel Tercih İlişkileri ile hesaplanmış ve Öztayşi ve diğ. (2015)'te önerdiği gibi normalize edilmiştir. Daha sonra, kriterlere uygun olan performans gösterge fonksiyonlarına göre performans skorları hesaplanmıştır. Uygulanan metodun adımları aşağıdaki gibidir:

1. Adım: Her bir uzman, performans değerlendirilirken kullanılan ana kriterlerden birini diğer her bir ana kriterle karşılaştırıp kabul edilebilir bir tamamlanmamış aralık değerli sezgisel tercih ilişkisini $\tilde{Q}$ oluşturur. Uzmanlar bu karşılaştırmayı yaparken Çizelge 4.1'de gösterilen dilsel ölçeği kullanırlar. Uzmanlardan sadece tek bir kriterin diğerleriyle karşılaştırılması istendiğinden ve kalan karşılaştırmalar hesaplama yöntemiyle bulunacağından, kriter karşılaştırmaları arasında bir tutarsızlık oluşmaz; ancak burada uzmanların tek seferde doğru bir karşılaştırma yaptıkları varsayılır.

Çizelge 4.1 : Dilsel ölçek ve karşılık gelen ADSBS'leri.

Dilsel Terimler	Üyelik & Üye Olmama Değerleri
Kesinlikle Düşük (AL)	([0.10, 0.25],[0.65, 0.75])
Çok Düşük (VL)	([0.15, 0.30],[0.60, 0.70])
Düşük (L)	([0.20, 0.35],[0.55, 0.65])
Orta Düşük (ML)	([0.25, 0.4]),[0.50, 0.60])
Yaklaşık Eşit (AE)	([0.45, 0.55],[0.30, 0.45])
Orta Yüksek (MH)	([0.50, 0.60],[0.25, 0.40])
Yüksek (H)	([0.55,0.65],[0.20, 0.35])
Çok Yüksek (VH)	([0.60,0.70],[0.15,0.30])
Kesinlikle Yüksek (AH)	([0.65,0.75],[0.10,0.25])

2. Adım: $\tilde{Q}$ 'nun bilinen elemanları ve (4.39) aritmetik ortalama denklemi kullanılarak her bir uzman için geliştirilmiş sezgisel tercih ilişkisi elde edilir. Geliştirilmiş Sezgisel Tercih İlişkisinde diyagonalde gösterilen, her bir kriterin kendisiyle karşılaştırması için, Tamamen Eşit (EE) dilsel terimine karşılık gelen ([0.50, 0.50], [0.50, 0.50]) ADSBS'si kullanılmıştır.

3. Adım: Her bir uzmanın geliştirilmiş tercih ilişkisindeki satırlarını birleştirmek için (4.41) denklemi kullanılır. Burada, $w_1 = w_2 = \ ... = w_n = 1/n$ kabul edilir.

4. Adım: Birleştirilmiş tercih değerlerini sıralayabilmek için (4.43)'teki denklem kullanılarak kriter bağıntıları skor değerlerine göre sıralanır.

5. Adım: (4.42) 'teki denklem yardımıyla sıralanmış değerlerin ağırlık vektörü $\mathrm{w}_G = \left(w_{\mathrm{G}_1}, w_{\mathrm{G}_2}, \ldots, w_{\mathrm{G}_n}\right)^T$ hesaplanır.

6. Adım: (4.41) denklemiyle tanımlanan ASBAB işleminde 5. adımda hesaplanan ağırlık vektörü kullanılarak kriterlerin ADSBS karşılıkları elde edilir.

7. Adım: Aşağıdaki netleştirme denklemi kullanılarak bulanık sayıların net karşılıkları elde edilir ve elde edilen bu değerler toplamlarının 1 olması için normalize edilir (Öztayşi ve diğ, 2016).

$$I(\tilde{q}) = \frac{a + b + (1 - c) + (1 - d) + a \times b - \sqrt{(1 - c) \times (1 - d)}}{4} \qquad (4.45)$$

Burada, $a = \tilde{\mu}_{ij}{}^{L}, \quad b = \tilde{\mu}_{ij}{}^{U}, \quad c = \tilde{v}_{ij}{}^{L} \quad ve \quad d = \tilde{\mu}_{ij}{}^{U}$ 'dir.

8. Adım: [1-6] arasındaki adımlar her bir ana kriter grubu için tekrar edilir.

9. Adım: Her bir alt kriterin (4.45) denklemiyle net karşılıkları elde edilir; ancak net karşılıklar normalize edilmezler; bunun yerine her bir alt kriter grubunun ağırlıklarının

toplamını, bağlı oldukları ana kriterin ağırlığına eşitlenir ve böylelikle tüm alt kriterlerin ağırlıklarının toplamı ana kriterlerde olduğu gibi 1'e eşit olur.

10. Adım: Uzmanlar her bir çalışan için gözlemlenen performans değerini [1-5] ölçeğinde belirtir.

11. Adım: Çalışanların her bir kriter için aldığı skor, kritere uygun performans gösterge fonksiyonuyla hesaplanır.

12. Adım: Her bir kriter için hesaplanan skor ağırlıklarla çarpılır ve ağırlıklandırılmış değerler toplanarak toplam performans skoru elde edilir.

5. UYGULAMA

5.1 Vaka Anlatımı

Hızlı değişen iş ortamları ve devrim niteliğindeki ilerici bilgi teknolojilerinin yazılım geliştirmeyi zorlaştırmaktadır (Barki ve diğ, 2001). Sürekli değişen iş ortamlarına ve sistem gereksinimlerine hızlı bir şekilde yanıt vermek yazılım geliştirme projelerinde kritik bir konudur (Gefen ve Ridings, 2002).

Bu noktada işletmeler çevik yazılım geliştirme metoduna yönelip fayda sağlasalar da, çevik metotlarda nicel kalite ölçümü yapmak, özellikle yazılım geliştiricilerin bireysel performansını değerlendirirken oldukça zordur. Yazılım geliştirme mühendislerinin performansına ilişkin değerlendirme kriterleri, geleneksel olarak çevik geliştirme ilkelerine uymayan metrikler tarafından belirlenir (Alnaji ve Salameh, 2015). Çevik geliştirme metotları, ekip işbirliği, mentorluk, ekip çalışması ve bilgi aktarımı gibi yazılım geliştirme dünyasında geleneksel olmayan özellik ve becerilere büyük önem vermektedir; ancak bu değerler yazılım geliştirme mühendislerinin değerlendirmesine yansıtılmamaktadır. Çevik geliştirme metotları, sosyal becerilere, yaratıcı düşünceye ve kendi kendine organize olmaya daha fazla vurgu yapılmasını gerektirirken, çoğu çevik yazılım metodunu kullanan işletmede, performans değerlendirme kriterleri hala teknik becerilere ve talimatlara uyabilme yeteneğine odaklanmaktadır. Sonuç olarak, performans değerlendirmesi genellikle ekip üyelerinin gerçek yeteneklerini yansıtmamaktadır (Alnaji ve Salameh, 2015).

Çevik yazılım geliştirme ortamlarında yazılım mühendislerinin performansını değerlendirmek karmaşık bir iştir; çünkü çevik yazılım geliştirme ortamları yazılım geliştirme mühendislerinin iyi bir kodlayıcı olmalarının yanı sıra; iyi iletişim kurabilen, iyi sunum yapabilen, iyi test yapabilen, güzel tasarımlar ortaya koyabilen ve yeterince ticari bilgiye sahip bir birey olmalarını gerektirmektedir. Dolayısıyla, geleneksel yazılım mühendisi performans değerlendirme teknikleri, çevik geliştirme ilkelerini ve ölçümlerini yansıtmaz (Alnaji ve Salameh, 2015).

Yazılım uygulamaları genellikle bireyler tarafından değil, ekipler tarafından yapılır. Her birey, yüksek kalitelide bir yazılım üretmek için katkı sağlar ve bu, bireylerin birbirlerine yardım etmeleri ve yüksek kaliteli yazılımlar sunmak için sorumluluğu paylaşmaları anlamına gelir. Dolayısıyla bireyler kahraman olmaya değil iyi bir yazılım ortaya çıkarmaya çalışırlar. Bu da çalışanların bireysel performansına odaklanan performans yönetim sistemiyle bir çatışma oluşturur (Al-Heyasi, 2018).

Birçok durumda, işletmelerin yapısı ve yönetim kuralları, ekiplerini tek bir birim olarak değerlendirme konusunda çok az özgürlük verir ya da hiç vermez. Dolayısıyla yöneticiler takım elemanlarını değerlendirirken çalışanların performansını birbirinden ayırma ve adil olma konusunda zorluk yaşamaktadırlar. Yöneticilerin biraz özgürlükleri olsa bile yazılım geliştirme ekiplerinde ekip olarak performansın değerlendirilmesi çok yaygın bir kavram değildir ve literatürde bu tür bir performans değerlendirmesi için yapılmış çalışmalar oldukça azdır (Narayan, 1996).

Performans ölçümü, çalışanları en iyisini yapmaları ve daha üretken olmaları için motive etmek ve onları geliştirmek için neler yapılabileceğini bilmek için gereklidir; ancak çevik ekipler, en iyi tasarımların, çözümlerin ve ürünün teslimatı sorumluluğunun bir proje yöneticisi tarafından değil tüm ekip üyeleri tarafından paylaşıldığı, kendi kendine organize olabilen ve sürekli iş birliği yapan ekiplerdir. Çevik metodu kullanan birçok işletme, geleneksel performans değerlendirme yöntemlerini kullanmaya devam ettiklerinden, bireylerinin performansını ölçmek için mücadele eder. Bireysel performans ölçümü, çevikliği önleyici bir uygulama olarak kabul edilir; çünkü bu işbirliği kavramına terstir ve ekip üyelerinin davranışlarını, değerlendirme sistemine uymak için değiştirmeye zorlar; bu da takımın etkinliğini en aza indirir (Al-Heyasi, 2018).

Günümüzün hızla değişen teknolojisine uyum sağlayabilmek için yazılım geliştirme ekiplerinin çevik metotlar uygulaması kaçınılmazdır. İşletmelerin, çalışanların gelişimini destekleyip, daha kaliteli bir şekilde daha verimli ürünlerin ortaya çıkmasını sağlamak ve maaş zammı, prim, terfi gibi konulara karar vermek için bireysel performans yönetim sistemlerini tercih etmesi de kaçınılmazdır. Dolayısıyla, çalışanların bireysel performansını değerlendirirken kullanılacak kriterlerin, çevik yazılım geliştirme ilkeleriyle çelişmeyecek şekilde belirlenmesi ve önemlerine göre ağırlıklandırılması gerekmektedir. Geleneksel performans değerlendirme yöntemlerinde kullanılan kriterler, sayısal verilerle daha kolay ifade edilebilirken,

çevik metot uygulayan ekiplerin performansını değerlendirmek için kullanılacak iş birliği, sosyal yetenek, koçluk, bilgi aktarımı, bilgi genişliği vb. soyut kriterleri sayısal verilerle ifade edebilmek ve nesnel bir şekilde karar verebilmek zordur. Bu zorluğun temel sebebi, soyut kriterlerin net sınırlarla ölçülebilir ve kesin olmamasıdır. Performans soyut kriterlere göre değerlendirilirken, doğal dilde kesin olmayan ifadeler veya sözcükler kullanılır (Ahmed ve diğ, 2013). Ayrıca, çalışanların performans değerlendirmesi, değerlendiricinin tecrübesinden, duyarlılığından ve standartlarından etkilenebilir. Dolayısıyla, değerlendiricinin verdiği puanlar sadece yaklaşık değerlerdir ve değerlendirmede içsel bir belirsizlik vardır (Arbaiy ve Suradi, 2007).

Ekip olarak daha başarılı sonuçlar alınabilmesi için ekip üyelerinin de günümüzün yazılım dünyasının gerekliliklerine uygun bir şekilde kendilerini geliştirmeleri gerekmektedir. Bu gerekliliklerden hangisinin bireysel ve ekip performansına daha fazla katkı sağlayacağını ölçmek, bu gerekliliklerin doğası gereği oldukça zordur; çünkü bu gerekliliklerin çoğunun kesin değerlerle ifade edilebilecek bir çıktısı yoktur. Ancak gelecekte daha başarılı olabilmek için takımın hangi gereklilikler üstünde yoğunlaşması gerektiğini bilmek, yani performans değerlendirmesi sırasında kullanılan kriterlerin birbirine göre önemini hesaplayabilmek ve sonuca göre mevcut durumun değerlendirmesini yapıp, geleceğe yönelik atılması gereken adımları planlayabilmek açısından önemlidir.

5.2 Temel Performans Göstergeleri

Bu çalışmada, telekomünikasyon sektöründe faaliyet gösteren uluslararası bir firmanın Ar-Ge departmanındaki bir yazılım test ekibinin üyelerinin performansını değerlendirmek için göz önünde bulunan kriterler incelenmiştir. 10 kişilik ekibin, her bir üyesinin performansı, iki farklı test ekibi yöneticisi tarafından her bir kriter için ayrı ayrı değerlendirilip, çalışanların aldıkları puan toplanarak elde edilen sonuca göre değerlendirilmektedir. Yöneticiler performans puanlarını Çizelge 5.1'de gösterildiği gibi başarılarına göre [1-5] ölçeğinde belirtmektedir.

Performans değerlendirmesi için uzmanlar tarafından teknik beceriler, sosyal beceriler, görev başarımı ve ekibe katkı olmak üzere 4 ana kriter belirlenmiştir. Çizelge 5.2'de de gösterildiği gibi her ana kriterin 3'er alt kriteri vardır. Kriterlerin tümü için uygulama sırasında, III nolu performans gösterge fonksiyonu kullanılmıştır.

Çizelge 5.1 : Performans değerlendirme ölçeği.

Puan	Dilsel karşılığı
1	Çok başarısız
2	Başarısız
3	Normal
4	Başarılı
5	Çok başarılı

Çizelge 5.2 : Ana kriterler ve alt kriterleri.

K1-Teknik beceriler		K2-Sosyal beceriler		K3-Görev başarımı		K4-Ekibe katkı	
K11	Teknik yeterlilik ve sorun çözme	K21	Genel Davranış	K31	Dürüstlük ve görevlerin tamamlanma süresi	K41	Ekip çalışmasına katkı
K12	Çıktı kalitesi	K22	Kendini adama	K32	Çoklu görevleri başarabilme	K42	Bilgi paylaşımı
K13	Kendini geliştirme	K23	Stres yönetimi	K33	Zorlu görevlerin başarımı	K43	Yönlendiricilik ve sorumluluk alma

5.2.1 Teknik beceriler

5.2.1.1 Teknik yeterlilik ve sorun çözme (K11)

Uygulamanın yapıldığı firmada, yazılım test mühendislerinden farklı test tipleri (Fonksiyonel, Kullanıcı Arayüzü, Güvenlik, Performans, Kurulum, Güncelleme vb.) için gerek manuel gerekse otomasyon test senaryolarını tasarlamaları ve yazılım teste hazır olduğunda, tasarlanan senaryoları denemeleri beklenmektedir. Test senaryolarını tasarlarken akışı doğru bir şekilde oluşturabilmek için sistematik düşünme ve algoritma kurabilme yetenekleri önemlidir. Aynı zamanda daha verimli çalışabilmek için, söz konusu testin tipine göre değişebilen farklı yazılım test araçlarını etkin bir şekilde kullanabilmeleri gerekir. Yazılım test ekibinin bir başka sorumluluğu da, çapraz fonksiyonel yeterlilik gerektiren bir iş olan, sunuculara test ortamını kurmak, test ortamını aktif ve güncel tutmak ve testin koşulmasına engel olabilecek herhangi bir sorunla karşılaşıldığında bu sorunu çözmektir. Bu nedenle, çalışanların farklı işletim sistemleri ve veritabanlarıyla kolayca çalışabilmelerini sağlayan güçlü bir teknik altyapıya sahip olmalarının yanı sıra analitik düşünebilme yetenekleri de önemlidir. Bu kriterin başarılıp başarılmadığını ölçmek için toplanabilecek sayısal veriler oldukça azdır ve maalesef tam anlamıyla başarımı göstermezler. Örneğin, çözülen sorunların sayısı bir gösterge olarak kullanılmak istense de, sorunu çözme yönteminin verimliliği ancak çözüm bir başkası tarafından gözden geçirildiğinde

anlaşılabilir ve sayısal bir veriyle değil doğal dilde kesin olmayan sözcüklerle ifade edilebilir.

5.2.1.2 Çıktı kalitesi (K12)

Yazılım test mühendislerinin en önemli çıktıları, oluşturdukları test senaryoları, özet test sonuç raporları ve test edilen uygulamadaki bulguların detaylı raporlarıdır. Kaliteli bir test senaryosu, test raporu ve bulgu raporunda, şablonlarda belirtilen unsurların hepsininin tamam olmasının yanı sıra; yazılan metnin açık, net ve her okuyanın aynı şekilde anlayacağı bir biçimde olması beklenir. Örneğin, bir bulgu raporunda sorunun oluşması için gerekli ön şartlar, senaryo adımları, beklenen sonuç, mevcut sonuç, sistem logları, ekran görüntüleri vb. unsurların belirtilmiş olması gerekir; ancak bu yeterli değildir. Bulguyu çözecek yazılım geliştiricisi, bulguyu okuduğunda sorunun ne olduğunu ve nasıl oluştuğunu net bir şekilde anlayabilmelidir. Aksi halde gereksiz zaman kayıpları yaşanır. Bu kriterin başarımını ölçmek için, çıktıları girdi olarak alan kişilerden gelen pozitif ve negatif geri bildirimlerin ve çıktıları gözden geçirme yazılım araçlarına girilen yorumların sayısının tasarlanan test senaryoları ve raporlanan bulguların sayısına oranı bir gösterge olarak kullanılabilir. Ancak uygulamanın yapıldığı firmada, her bir çalışan için bu veriyi otomatik olarak toplayacak bir araç bulunmadığından, veriyi toplamak oldukça zahmetli bir iştir ve yöneticilerin genel gözlemine dayalı bir değerlendirme yapılmaktadır.

5.2.1.3 Kendini geliştirme (K13)

Hızla değişen ve gelişen yazılım dünyasında, yeni çıkan teknoloji ve metotları öğrenip kullanmaya başlamak başarılı olmak için zorunludur. Aksi halde, müşterilerin gereksinimlerini daha iyi ve verimli teknolojilerle karşılayabilecek rakiplerle mücadele etmek zorlaşır. Bu hızlı değişime kolay uyum sağlayabilmek için, uygulamanın yapıldığı ekip de çevik yöntemleri tercih etmektir; ancak ekibin çevik olabilmesi için ekip üyelerinin de hızlı öğrenme ve uyum sağlayabilme yeteneklerine sahip olması gerekir. Ek olarak, işe alınan her çalışanın, atandığı projenin gerektirdiği tüm alanlarında yüksek bilgi seviyesine sahip olmasını beklemek gerçekçi değildir. Bu nedenle çalışanlardan atandıkları projenin gereksinimlerine uygun olarak kendilerini geliştirmeleri beklenir. Bu kriteri başarımın ölçümünü sayısal verilerle yapmak çok zordur.

5.2.2 Sosyal beceriler

5.2.2.1 Genel davranış (K21)

Daha önce de bahsedildiği gibi yazılım geliştirme bir ekip işidir ve üyeleri ekiple uyum içerisinde çalışabilmeli ve genel motivasyonu düşürecek davranışlardan kaçınmalıdır.

5.2.2.2 Kendini adama (K22)

Kendini adama, herhangi bir şeye, önemli olduğu için çok fazla enerji verme isteği ve değer katmak için harcanan çaba olarak tanımlanabilir. Uygulamanın yapıldığı global firma da temel değerlerinden biri olarak kendini adamayı seçmiştir. Çalışanların, görevlerini sahiplendikleri ve karşılarına çıkan zorluklarla mücadele etmekten yılmadıkları takdirde başarıya ulaşacaklarına inanılır. Bu kriter, bazen zorlu problemlere çözüm aramayı, bazen de aynı test senaryolarını farklı yazılım sürümleri için tekrar tekrar koşmayı gerektiren yazılım test mühendisliği için de önemlidir. Bu kriteri başarımın ölçümünü sayısal verilerle yapmak çok zordur.

5.2.2.3 Stres yönetimi (K23)

Bir yazılım test mühendisi, birçok durumda kendini stresle mücadele etmek zorunda bulabilir. Bu durumlarla başa çıkabilmek için doğru iletişim teknikleri ve öz motivasyon önemlidir. Örneğin, tespit edilen bir bulguyu kabul etmek istemeyen bir yazılım geliştiricisine senaryoda beklenen sonucu düzgün bir şekilde anlatabilmeli veya müşteriyle birlikte yapılan Kullanıcı Kabul Testleri'nde müşteriden gelen yorum ve isteklere profesyonel bir biçimde ve ekibin çıkarlarıyla çelişmeyecek bir şekilde cevap verebilmelidir. Bir başka konu da test mühendislerinin üstündeki zaman baskısıdır. Ürünün tesliminden önceki son aşama test aşaması olduğundan, müşteriyle ürünün teslim tarihi konusunda tekrar pazarlık yapılamıyorsa ürünün geliştirilmesi süresince çıkan herhangi bir sorun test süresini kısaltmaktadır ve bu da ürünü söz verilen teslim tarihine yetiştirebilmek için test mühendislerinin üstünde zaman baskısı oluşturur. Bu baskının konsantrasyon kaybına yol açıp bulguların yakalanmasına engel olmasını önlemek için stresle baş edebilmek önemlidir. Bu kriterin başarımını ölçmek için de toplanabilecek sayısal bir gösterge yoktur.

5.2.3 Görev başarımı

5.2.3.1 Dürüstlük ve görevlerin tamamlanma süresi (K31)

Scrum ilkelerine göre Scrum ekibindeki kişilere ayrı ayrı görev ataması yapılmaz. Kişiler görev havuzundan kendileri için uygun olduğunu düşündükleri görevleri alırlar; ancak planlama toplantılarında konan hedefe ulaşabilmek için kişilerin verimli bir şekilde çalışıp, üstlendikleri görevi tecrübe ve yeteneklere göre en kısa zamanda tamamlamaları beklenir. O Sprint için planlanan tüm işlerin tamamlanması için alt görevlerin hepsinin başarılması gerektiğinden, her bir çalışanın üstlendiği görevi dürüstlük içinde tamamlayıp yeni bir görev alması gerekir. Bir görevin tamamlanma süresi görevi üstlenen kişinin bilgi ve tecrübesine göre değişeceğinden, çalışanın performansını değerlendiren yönetici hem görevlerin kapsam ve zorluk derecelerini hem de çalışanın yeterliliklerini iyi gözlemlemelidir.

5.2.3.2 Çoklu görevleri başarabilme (K32)

Bu kriter, test senaryolarını yazma ve testleri koşma gibi görevlerin yanı sıra, genelde acil bir şekilde çözülmesi gereken bir sorun olduğunda (test ortamlarının işlevselliğini yitirmesi ya da ekip arkadaşının desteğe ihtiyacı olması gibi) çalışanın birden fazla görevi aynı zaman diliminde nasıl yürüttüğünü, önceliklendirmeyi ve planlamayı nasıl yaptığını değerlendirmek için kullanılmaktadır.

5.2.3.3 Zorlu görevlerin başarımı (K33)

Uygulamanın yapıldığı firmada yazılım test mühendislerinin yurt içi ve yurt dışı saha destek görevleri ya da sunucuların bulunduğu laboratuvarlarda yapılacak kurulumların takibi ve yönetimi gibi Scrum ekibinin üstünde çalıştığı işlere ek olarak başka zorlu görevleri olabilmektedir. Bu kriter bu görevlerdeki başarım oranını değerlendirmek için kullanılmaktadır.

5.2.4 Ekibe katkı

5.2.4.1 Ekip çalışmasına katkı (K41)

Daha önce çevik yöntemlerin ve Scrum'ın ilkelerini açıklarken bahsedildiği gibi, bir yazılım ürününü geliştirirken ürünün zamanında ve kaliteli bir şekilde teslim

edilmesindeki sorumluluk tüm takıma aittir. Her ekip üyesinin güçlü ve zayıf olduğu alanlar olabilir. Bir ekip üyesinin zorluk yaşadığı bir durumda yardımcı olmak, farklı çözüm önerileri sunmak ve işbirliği yapmak takımın başarısını artırır. Ekip çalışmasına katkı, bir sorunun çözümünde sorunu yaşayan kişiyle beraber çalışarak çözüme gitmek olabileceği gibi, öğrenilen yeni ve daha iyi bir yöntemin ekiple paylaşılması veya her Sprint sonunda yapılan Retrospection (Geriye bakma) toplantılarında ekibi geliştirecek pozitif geri bildirimlerde bulunulması olabilir. Birçok firmada olduğu gibi uygulamanın yapıldığı firmada da çalışanların firmada bulundukları süre boyunca edindikleri bilgi ve deneyimi firma içinde korunacak bir değere dönüştürüp kalıcı hale getirmeleri ve bu değeri paylaşmaları, bilginin tekrar kullanılabilmesi ve olası zaman kayıplarını engellemek açısından önemlidir. Bu aynı zamanda, ekipteki diğer çalışanların kendilerini geliştirmeleri açısından faydalıdır. Bu yüzden yazılım test ekibi çalışanlarından da firmanın bilgi paylaşım ortamlarına içerik yüklemeleri ve/veya ekip içi eğitim/bilgi paylaşımı oturumları düzenlemeleri beklenmektedir.

5.2.4.2 Bilgi paylaşımı (K42)

Birçok firmada olduğu gibi uygulamanın yapıldığı firmada da çalışanların firmada bulundukları süre boyunca edindikleri bilgi ve deneyimi firma içinde korunacak bir değere dönüştürüp kalıcı hale getirmeleri ve bu değeri paylaşmaları, bilginin tekrar kullanılabilmesi ve olası zaman kayıplarını engellemek açısından önemlidir. Bu aynı zamanda, ekipteki diğer çalışanların kendilerini geliştirmeleri açısından faydalıdır. Bu yüzden yazılım test ekibi çalışanlarından da firmanın bilgi paylaşım ortamlarına içerik yüklemeleri ve/veya ekip içi eğitim/bilgi paylaşımı oturumları düzenlemeleri beklenmektedir. Bu kriteri başarımı ölçmek için paylaşılan içeriklerin ve düzenlenen oturumların sayısı kullanılabilir.

5.2.4.3 Yönlendiricilik ve sorumluluk alma (K43)

Scrum ekibinin planladığı işlerin alt görevlerinin zorluk dereceleri birbirinden farklıdır ve çalışanların zorlu görevleri üstlenme sorumluluğundan kaçmaması beklenir. Ayrıca Scrum ekibi, müşterinin isteklerini yerine getirecek yazılımın tasarlanması sürecinde birçok noktada ekip olarak karar verse de bu kararın alınmasında yönlendirici rol oynamak, yeni önerilerde bulunmak ve araştırma yapmak ek sorumluluk almayı gerektirir. Bu kriteri başarımın ölçümünü de sayısal verilerle yapmak çok zordur.

5.3 Geliştirilen Yazılım

Uygulamada kullanılan metodun [2-7] arasındaki adımlarını daha hızlı ve basit bir şekilde gerçekleyebilmek için Java programlama diliyle bir yazılım geliştirilmiştir. Bu yazılım, 1. adımda uzmanların dilsel terimlerle belirlediği, bir kriterin diğer kriterlerle olan tercih ilişkisini, Excel dosya formatında Şekil 5.1'deki örnekte görüldüğü gibi girdi olarak alır. Excel dosyasında uzmanların seçebilecekleri dilsel terimler, Çizelge 4.1'deki değerlerle sınırlandırılmıştır.

	Teknik beceriler	Sosyal beceriler	Görev başarımı	Ekibe katkı
Uzman 1	EE	H	MH	MH
Uzman 2	EE	AE	AE	MH
Uzman 3	EE	H	MH	VH

Şekil 5.1 : Yazılımın Excel girdi örneği.

Yazılım, Excel'deki satır sayısından uzman sayısını, sütun sayısından da kriter sayısını otomatik olarak hesaplar. Daha sonra, [2-7] adımlarında belirtilen denklemleri sırasıyla kullanarak kriterlerin normalize edilmiş ağırlık değerlerini Şekil 5.2'deki gibi çıktı olarak ekrana basar.

```
Kriter No    ADSBS Ağırlıkları                    Netleştirilmiş Ağırlıklar    Normalize edilmiş
C1           ([0,511, 0,589],[0,282, 0,411])      0,514                        0,306
C2           ([0,375, 0,470],[0,439, 0,530])      0,385                        0,229
C3           ([0,396, 0,492],[0,416, 0,508])      0,406                        0,241
C4           ([0,367, 0,462],[0,448, 0,538])      0,377                        0,224
```

Şekil 5.2 : Yazılımın çıktı örneği.

Yazılım Çizelge 5.3'te gösterildiği gibi 6 Java sınıfından oluşmaktadır. Yazılımın detaylarına EK A kısmından ulaşılabilir.

Çizelge 5.3 : Yazılımdaki Java sınıfları.

Sınıf	Amacı
App.java	Bu sınıf uygulamada kullanılan adımların işlendiği, yazılımın ana sınıfıdır. Bu sınıfta, Excel dosyasındaki veriler okunup, [2-7] arasındaki adımları tek tek uygulamak için diğer sınıflarda tanımlanan metot ve nesneler kullanılmış ve sonuç ekrana bastırılmıştır.
IVIFN.java	Bu sınıfta, ADSBS nesnesi ve $\tilde{\mu}_{ij}{}^{L}$, $\tilde{\mu}_{ij}{}^{U}$, $\tilde{\nu}_{ij}{}^{L}$, $\tilde{\mu}_{ij}{}^{U}$ elemanları tanımlanmıştır.
LinguisticTerms.java	Bu sınıfta, Çizelge 4.1'deki dilsel terimlerin ADSBS karşılıkları tanımlanmıştır.

Çizelge 5.3 (devam): Yazılımdaki Java sınıfları.

Sınıf	Amacı
Experts.java	Bu sınıfta, her bir uzman için ayrı ayrı hesaplanan, Geliştirilmiş Sezgisel Tercih İlişkisi ve birleştirilmiş tercih değerlerini elde etmek için kullanılacak matris ve liste tanımlanmıştır.
Equations.java	Bu sınıfta [2-7] adımlarında kullanılan her bir denklem ayrı bir metot olarak tanımlanmıştır.
SortedPreferenceRelation.java	Bu sınıfta, 4. adımda bahsedilen birleştirilmiş tercih değerlerini sıralayabilmek için kullanılan metot tanımlanmıştır.

5.4 Uygulama Adımları

1. Adım: Bu adımda Ar-Ge merkezindeki iki test takım lideri ve bir proje yöneticisi tarafından 1. ana kriterin (K1) diğer kriterlere göre önemi Çizelge 5.4'te gösterildiği gibi dilsel ifadelerle belirlenmiştir.

Çizelge 5.4 : K1 kriterinin diğer kriterlere göre uzmanlar tarafından belirlenen önemi.

		K1	K2	K3	K4
Uzman 1	K1	EE	H	MH	L
Uzman 2	K1	EE	AE	ML	ML
Uzman 3	K1	EE	VH	H	ML

2. Adım: Bu adımda, her bir uzman için ikili karşılaştırma matrisleri elde edilmiştir. Çizelge 5.5, 5.6 ve 5.7'de (4.39) aritmetik ortalama denklemi kullanılarak elde edilen elemanlar koyu şekilde gösterilmiştir. Örneğin $\tilde{q}_{23}$ değeri aşağıdaki gibi hesaplanmıştır:

$$\tilde{q}_{23} = \frac{\tilde{q}_{21} \oplus \tilde{q}_{13}}{2}$$

$$= \frac{([0{,}200, 0{,}350], [0{,}550, 0{,}650]) \oplus ([0{,}500, 0{,}600], [0{,}250, 0{,}400])}{2}$$

$$= ([0{,}350, 0{,}475], [0{,}400, 0{,}525])$$

3. Adım: Her bir uzman için geliştirilmiş tercih ilişkisindeki satırlar Çizelge 5.8'deki gibi birleştirilmiştir.

Çizelge 5.5 : Uzman 1'in tamamlanmış ikili karşılaştırma matrisi.

Uzm 1	K1	K2	K3	K4
K1	([0,500, 0,500],[0,500, 0,500])	([0,550, 0,650],[0,200, 0,350])	([0,500, 0,600],[0,250, 0,400])	([0,500, 0,600],[0,250, 0,400])
K2	([0,200, 0,350],[0,550, 0,650])	([0,500, 0,500],[0,500, 0,500])	**([0,350, 0,475],[0,400, 0,525])**	**([0,350, 0,475],[0,400, 0,525])**
K3	([0,250, 0,400],[0,500, 0,600])	**([0,400, 0,525],[0,350, 0,475])**	([0,500, 0,500],[0,500, 0,500])	**([0,375, 0,500],[0,375, 0,500])**
K4	([0,250, 0,400],[0,500, 0,600])	**([0,400, 0,525],[0,350, 0,475])**	**([0,375, 0,500],[0,375, 0,500])**	([0,500, 0,500],[0,500, 0,500])

Çizelge 5.6 : Uzman 2'nin tamamlanmış ikili karşılaştırma matrisi.

Uzm 2	K1	K2	K3	K4
K1	([0,500, 0,500],[0,500, 0,500])	([0,450, 0,550],[0,300, 0,450])	([0,450, 0,550],[0,300, 0,450])	([0,500, 0,600],[0,250, 0,400])
K2	([0,300, 0,450],[0,450, 0,550])	([0,500, 0,500],[0,500, 0,500])	**([0,375, 0,500],[0,375, 0,500])**	**([0,400, 0,525],[0,350, 0,475])**
K3	([0,300, 0,450],[0,450, 0,550])	**([0,375, 0,500],[0,375, 0,500])**	([0,500, 0,500],[0,500, 0,500])	**([0,400, 0,525],[0,350, 0,475])**
K4	([0,250, 0,400],[0,500, 0,600])	**([0,350, 0,475],[0,400, 0,525])**	**([0,350, 0,475],[0,400, 0,525])**	([0,500, 0,500],[0,500, 0,500])

Çizelge 5.7 : Uzman 3'ün tamamlanmış ikili karşılaştırma matrisi.

Uzm 3	K1	K2	K3	K4
K1	([0,500, 0,500],[0,500, 0,500])	([0,550, 0,650],[0,200, 0,350])	([0,500, 0,600],[0,250, 0,400])	([0,600, 0,700],[0,150, 0,300])
K2	([0,200, 0,350],[0,550, 0,650])	([0,500, 0,500],[0,500, 0,500])	**([0,350, 0,475],[0,400, 0,525])**	**([0,400, 0,525],[0,350, 0,475])**
K3	([0,250, 0,400],[0,500, 0,600])	**([0,400, 0,525],[0,350, 0,475])**	([0,500, 0,500],[0,500, 0,500])	**([0,425, 0,550],[0,325, 0,450])**
K4	([0,150, 0,300],[0,600, 0,700])	**([0,350, 0,475],[0,400, 0,525])**	**([0,325, 0,450],[0,425, 0,550])**	([0,500, 0,500],[0,500, 0,500])

Örneğin, Uzman 1-K1 birleştirilmiş değeri aşağıdaki gibi hesaplanmıştır:

$$1 - [(1 - 0{,}500)(1 - 0{,}550)(1 - 0{,}500)(1 - 0{,}500)]^{1/4} = 0{,}513$$
$$1 - [(1 - 0{,}500)(1 - 0{,}650)(1 - 0{,}600)(1 - 0{,}600)]^{1/4} = 0{,}591$$
$$[(0{,}500)(0{,}200)(0{,}250)(0.250)]^{1/4} = 0{,}281$$
$$[(0500)(0{,}350)(0{,}400)(0{,}400)]^{1/4} = 0{,}409$$

Çizelge 5.8 : Uzmanların birleştirilmiş tercih değerleri.

	Uzman 1	Uzman 2	Uzman 3
K1	([0,513, 0,591],[0,281, 0,409])	([0,476, 0,551],[0,326, 0,449])	([0,539, 0,619],[0,247, 0,381])
K2	([0,359, 0,453],[0,458, 0,547])	([0,398, 0,494],[0,415, 0,506])	([0,372, 0,466],[0,443, 0,534])
K3	([0,388, 0,483],[0,426, 0,517])	([0,398, 0,494],[0,415, 0,506])	([0,400, 0,497],[0,411, 0,503])
K4	([0,388, 0,483],[0,426, 0,517])	([0,369, 0,464],[0,447, 0,536])	([0,343, 0,436],[0,475, 0,564])

4. Adım: Birleştirilmiş değerlerin skorları Çizelge 5.9'da gösterilmiştir.

Çizelge 5.9 : Uzmanların birleştirilmiş değerlerinin skorları.

		Birleştirilmiş Değer	Skor
K1	Uzman 1	([0,513, 0,591],[0,281, 0,409])	0,207
	Uzman 2	([0,476, 0,551],[0,326, 0,449])	0,126
	Uzman 3	([0,539, 0,619],[0,247, 0,381])	0,265
K2	Uzman 1	([0,359, 0,453],[0,458, 0,547])	-0,097
	Uzman 2	([0,398, 0,494],[0,415, 0,506])	-0,014
	Uzman 3	([0,372, 0,466],[0,443, 0,534])	-0,069
K3	Uzman 1	([0,388, 0,483],[0,426, 0,517])	-0,036
	Uzman 2	([0,398, 0,494],[0,415, 0,506])	-0,014
	Uzman 3	([0,400, 0,497],[0,411, 0,503])	-0,008
K4	Uzman 1	([0,388, 0,483],[0,426, 0,517])	-0,036
	Uzman 2	([0,369, 0,464],[0,447, 0,536])	-0,075
	Uzman 3	([0,343, 0,436],[0,475, 0,564])	-0,130
	Uzman 2	([0,513, 0,591],[0,281, 0,409])	0,207
	Uzman 3	([0,476, 0,551],[0,326, 0,449])	0,126

Örneğin K1 için skor değerleri aşağıdaki gibi hesaplanmıştır:

$$S(K1_1) = \frac{0{,}513 - 0{,}281 + 0{,}591 - 0{,}409}{2} = 0{,}207\ (2.sıra)$$
$$S(K1_2) = \frac{0{,}476 - 0{,}326 + 0{,}551 - 0{,}449}{2} = 0126\ (3.sıra)$$
$$S(K1_3) = \frac{0{,}539 - 0{,}247 + 0{,}619 - 0{,}381}{2} = 0{,}265\ (1.sıra)$$

Skorlara göre sıralanmış birleştirilmiş değerler Çizelge 5.10'daki gibidir.

5. Adım: Uzman sayısı 3 olduğu için $w_G = \left(w_{G_1}, w_{G_2}, w_{G_3}\right)^T$ Çizelge 5.11'deki gibi hesaplanmıştır.

Çizelge 5.10 : Skorlara göre sıralanmış birleştirilmiş değerler.

	1	2	3
K1	([0,539, 0,619],[0,247, 0,381])	([0,513, 0,591],[0,281, 0,409])	([0,476, 0,551],[0,326, 0,449])
K2	([0,398, 0,494],[0,415, 0,506])	([0,372, 0,466],[0,443, 0,534])	([0,359, 0,453],[0,458, 0,547])
K3	([0,400, 0,497],[0,411, 0,503])	([0,398, 0,494],[0,415, 0,506])	([0,388, 0,483],[0,426, 0,517])
K4	([0,388, 0,483],[0,426, 0,517])	([0,369, 0,464],[0,447, 0,536])	([0,343, 0,436],[0,475, 0,564])

Çizelge 5.11 : SAB ağırlık vektörü.

Vektör indisi	Ağırlık değeri
w_{G_1}	0,242895
w_{G_2}	0,514209
w_{G_3}	0,242895

6. Adım: Her bir kriterin bulanık ağırlığı Çizelge 5.12'de gösterilmiştir. Örneğin K1 için hesaplama aşağıdaki gibi yapılmıştır:

$$1-[(1-0{,}539)^{0{,}243}(1-0{,}513)^{0{,}514}(1-0{,}476)^{0{,}243}]=0{,}511$$
$$1-[(1-0{,}619)^{0{,}243}(1-0{,}591)^{0{,}514}(1-0{,}551)^{0{,}243}]=0{,}589$$
$$(0{,}247)^{0{,}243}(0{,}281)^{0{,}514}(0{,}326)^{0{,}243}=0{,}282$$
$$(0{,}381)^{0{,}243}(0{,}409)^{0{,}514}(0{,}449)^{0{,}243}=0{,}411$$

Çizelge 5.12 : Ana kriterlerin bulanık ağırlıkları.

	ADSBS Ağırlıkları
K1	([0,511, 0,589],[0,282, 0,411])
K2	([0,375, 0,470],[0,439, 0,530])
K3	([0,396, 0,492],[0,416, 0,508])
K4	([0,367, 0,462],[0,448, 0,538])

7. Adım: Çizelge 5.13'te kriter ağırlıklarının net karşılıkları ve normalize edilmiş değerleri gösterilmiştir. Örneğin K1 bulanık ağırlık değerinin net karşılığı aşağıdaki gibi hesaplanmıştır:

$$\frac{0{,}511+0{,}589+(1-0{,}282)+(1-0{,}411)+0{,}511\times0{,}589-\sqrt{(1-0{,}282)\times(1-0{,}411)}}{4}$$
$$=0{,}514$$

Çizelge 5.14'te ağırlıklarına göre azalan şekilde sıralanmış ana kriterler verilmiştir.

Çizelge 5.13 : Kriterlerin netleştirilmiş ve normalize edilmiş ağırlıkları.

	Netleştirilmiş Ağırlıklar	Normalize Edilmiş Ağırlıklar
K1	0,514	0,306
K2	0,385	0,229
K3	0,406	0,241
K4	0,377	0,224

Çizelge 5.14 : Ağırlıklarına göre sıralanmış ana kriterler.

Kriter No	Kriter adı	Normalize Edilmiş Ağırlıklar
K1	Teknik beceriler	0,306
K3	Görev başarımı	0,241
K2	Sosyal beceriler	0,229
K4	Ekibe katkı	0,224

8. Adım: Bu adımda [1-6] arasındaki adımlar alt kriter grupları için tekrar edilmiştir. Çizelge 5.15, 5.16, 5.17 ve 5.18'de uzmanların alt kriterleri, kendi grupları içinde değerlendirme sonuçları verilmiştir.

Çizelge 5.15 : K11 kriterinin diğer alt kriterlere göre uzmanlar tarafından belirlenen önemi.

		K11	K12	K13
Uzman 1	K11	EE	H	MH
Uzman 2	K11	EE	AE	ML
Uzman 3	K11	EE	VH	H

Çizelge 5.16 : K21 kriterinin diğer alt kriterlere göre uzmanlar tarafından belirlenen önemi.

		K21	K22	K23
Uzman 1	K21	EE	ML	MH
Uzman 2	K21	EE	ML	MH
Uzman 3	K21	EE	MH	MH

Çizelge 5.17 : K31 kriterinin diğer alt kriterlere göre uzmanlar tarafından belirlenen önemi.

		K31	K32	K33
Uzman 1	K31	EE	MH	MH
Uzman 2	K31	EE	AH	VH
Uzman 3	K31	EE	MH	ML

Çizelge 5.18 : K41 kriterinin diğer alt kriterlere göre uzmanlar tarafından belirlenen önemi.

		K11	K12	K13
Uzman 1	K41	EE	VH	MH
Uzman 2	K41	EE	AE	MH
Uzman 3	K41	EE	MH	MH

Çizelge 5.19'da tüm alt kriterlerin bulanık karşılıkları verilmiştir. Tablo sayısı çokluğundan dolayı her ana kriter grubu için tekrar edilen adımlardan [2-5] arasındakilerin sonuçlarına Ek B'de yer verilmiştir.

Çizelge 5.19 : Alt kriterlerin bulanık ağırlıkları.

	ADSBS Ağırlıkları
K11	([0,437, 0,518],[0,378, 0,482])
K12	([0,515, 0,586],[0,295, 0,414])
K13	([0,325, 0,415],[0,522, 0,585])
K21	([0,446, 0,523],[0,375, 0,477])
K22	([0,474, 0,547],[0,344, 0,453])
K23	([0,354, 0,444],[0,488, 0,556])
K31	([0,507, 0,581],[0,297, 0,419])
K32	([0,352, 0,440],[0,490, 0,560])
K33	([0,410, 0,491],[0,419, 0,509])
K41	([0,505, 0,575],[0,307, 0,425])
K42	([0,379, 0,464],[0,459, 0,536])
K43	([0,386, 0,471],[0,451, 0,529])

9. Adım: Çizelge 5.20'de tüm alt kriterlerin net karşılıkları ve toplamları bağlı oldukları ana kriterin normalize edilmiş ağırlığına eşit olacak şekilde oranlanmış ağırlıkları verilmiştir.

Çizelge 5.21'de ise alt kriterlerin ağırlıklarına göre sıralamaları verilmiştir.

10. Adım: Bu adımda her iki test takım lideri her bir çalışanı kriterlere göre değerlendirmiş Çizelge 5.1'de verilen dilsel karşılıklarına göre bir puan vermiştir. Ancak, eğer çalışanlara performans değerlendirme dönemi boyunca, K33 – Zorlu görevlerin başarımı kriteri çerçevesinde değerlendirilebilecek bir görev verilmediyse 0 puan verilmiştir. Her iki test grup liderinin de değerlendirmesi eşit derecede önemli olduğundan, takım liderlerinin verdikleri puanların ortalaması alınarak her bir çalışanın her bir kriter için aldığı puan belirlenmiştir. Çizelge 5.22'de çalışanların kriterlere göre aldıkları puanlar gösterilmiştir.

Çizelge 5.20 : Alt kriter ağırlıklarının net karşılıkları ve ana kriter ağırlığına göre oranlanmış ağırlıkları.

	Netleştirilmiş Ağırlıklar	Ana kriter ağırlığına göre oranlanmış ağırlıklar
K11	0,439	0,439
K12	0,513	0,513
K13	0,331	0,331
K21	0,445	0,445
K22	0,471	0,471
K23	0,359	0,359
K31	0,507	0,507
K32	0,356	0,356
K33	0,41	0,41
K41	0,502	0,502
K42	0,381	0,381
K43	0,387	0,387

Çizelge 5.21 : Ağırlıklarına göre sıralanmış alt kriterler.

	Netleştirilmiş Ağırlıklar	Ana kriter ağırlığına göre oranlanmış ağırlıklar
K12	Çıktı kalitesi	0,122
K11	Teknik yeterlilik ve sorun çözme	0,105
K31	Dürüstlük ve görevlerin tamamlanma süresi	0,096
K41	Ekip çalışmasına katkı	0,089
K22	Kendini adama	0,085
K21	Genel Davranış	0,080
K13	Kendini geliştirme	0,079
K33	Zorlu görevlerin başarımı	0,078
K43	Yönlendiricilik ve sorumluluk alma	0,068
K32	Çoklu görevleri başarabilme	0,067
K42	Bilgi paylaşımı	0,067
K23	Stres yönetimi	0,064

11. Adım: Bu adımda, K33 – Zorlu görevlerin başarımı alt kriteri için V, diğer kriterler için III nolu performans gösterge fonksiyonu kullanılarak çalışanların her bir kriter için aldığı skor hesaplanmıştır.

Tip V nolu fonksiyon için, a eşik değeri 2, b eşik değeri 5 olarak alınmış ve a eşik değerinin altındaki gösterge değerleri için performans skoru 0 olarak değerlendirilmiş, diğer gösterge değerleri için performans skoru $\frac{x_i}{5}$ formülüyle hesaplanmıştır. hesaplanmıştır.

Çizelge 5.22 : Çalışanların kriterlere göre aldıkları puanlar.

	Takım Lideri	K11	K12	K13	K21	K22	K23	K31	K32	K33	K41	K42	K43
	TL 1	5	5	3	3	3	4	3	5	0	2	4	3
Çalışan 1	TL 2	4	4	3	4	2	4	3	4	0	3	4	3
	ortalama	**4,5**	**4,5**	**3**	**3,5**	**2,5**	**4**	**3**	**4,5**	**0**	**2,5**	**4**	**3**
	TL 1	3	3	3	5	5	5	4	4	4	4	3	5
Çalışan 2	TL 2	3	3	4	5	5	5	4	5	5	5	3	4
	ortalama	**3**	**3**	**3,5**	**5**	**5**	**5**	**4**	**4,5**	**4,5**	**4,5**	**3**	**4,5**
	TL 1	3	4	3	5	3	4	3	4	0	5	3	4
Çalışan 3	TL 2	4	5	3	5	4	4	4	4	0	5	3	3
	ortalama	**3,5**	**4,5**	**3**	**5**	**3,5**	**4**	**3,5**	**4**	**0**	**5**	**3**	**3,5**
	TL 1	4	4	5	5	3	4	4	4	4	5	4	4
Çalışan 4	TL 2	4	4	4	5	4	4	4	5	5	5	3	5
	ortalama	**4**	**4**	**4,5**	**5**	**3,5**	**4**	**4**	**4,5**	**4,5**	**5**	**3,5**	**4,5**
	TL 1	5	5	5	5	5	5	5	5	0	5	4	4
Çalışan 5	TL 2	4	5	5	5	5	5	5	5	0	5	4	4
	ortalama	**4,5**	**5**	**5**	**5**	**5**	**5**	**5**	**5**	**0**	**5**	**4**	**4**
	TL 1	4	5	5	4	5	4	5	5	0	5	5	5
Çalışan 6	TL 2	5	5	5	4	5	4	5	5	0	5	5	5
	ortalama	**4,5**	**5**	**5**	**4**	**5**	**4**	**5**	**5**	**0**	**5**	**5**	**5**
	TL 1	4	4	5	5	4	4	4	4	4	4	3	5
Çalışan 7	TL 2	4	3	4	5	4	5	3	4	4	5	3	4
	ortalama	**4**	**3,5**	**4,5**	**5**	**4**	**4,5**	**3,5**	**4**	**4**	**4,5**	**3**	**4,5**
	TL 1	2	3	1	4	2	2	2	3	2	2	2	3
Çalışan 8	TL 2	2	3	2	3	2	2	2	2	2	3	2	3
	ortalama	**2**	**3**	**1,5**	**3,5**	**2**	**2**	**2**	**2,5**	**2**	**2,5**	**2**	**3**
	TL 1	2	2	2	3	2	2	2	2	2	3	2	2
Çalışan 9	TL 2	2	2	3	4	2	3	2	2	2	3	2	2
	ortalama	**2**	**2**	**2,5**	**3,5**	**2**	**2,5**	**2**	**2**	**2**	**3**	**2**	**2**
	TL 1	2	3	3	4	2	3	2	2	2	3	2	2
Çalışan 10	TL 2	2	3	3	5	2	4	2	2	3	3	2	3
	ortalama	**2**	**3**	**3**	**4,5**	**2**	**3,5**	**2**	**2**	**2,5**	**3**	**2**	**2,5**

Tip III nolu fonksiyon için, 5 gösterge değeri için performans skoru 1 olarak kabul edilmiştir. Diğer gösterge değerleri için performans skoru $\frac{x_i}{5}$ formülüyle hesaplanmıştır. Çizelge 5.23'te çalışanların kriterlere göre aldıkları performans skorları gösterilmiştir.

12. Adım: Kriter ağırlıklarıyla çarpımların toplamıyla elde edilen toplam performans skorları her bir çalışan için hesaplanmış ve azalan şekilde sıralama Çizelge 5.24'te gösterilmiştir.

Çizelge 5.23 : Çalışanların performans göstege fonksiyonuna göre hesaplanmış performans skorları.

	K11	K12	K13	K21	K22	K23	K31	K32	K33	K41	K42	K43
Çalışan 1	0,9	0,9	0,6	0,7	0,5	0,8	0,6	0,9	0	0,5	0,8	0,6
Çalışan 2	0,6	0,6	0,7	1	1	1	0,8	0,9	0,9	0,9	0,6	0,9
Çalışan 3	0,7	0,9	0,6	1	0,7	0,8	0,7	0,8	0	1	0,6	0,7
Çalışan 4	0,8	0,8	0,9	1	0,7	0,8	0,8	0,9	0,9	1	0,7	0,9
Çalışan 5	0,9	1	1	1	1	1	1	1	0	1	0,8	0,8
Çalışan 6	0,9	1	1	0,8	1	0,8	1	1	0	1	1	1
Çalışan 7	0,8	0,7	0,9	1	0,8	0,9	0,7	0,8	0,8	0,9	0,6	0,9
Çalışan 8	0,4	0,6	0,3	0,7	0,4	0,4	0,4	0,5	0	0,5	0,4	0,6
Çalışan 9	0,4	0,4	0,5	0,7	0,4	0,5	0,4	0,4	0	0,6	0,4	0,4
Çalışan 10	0,4	0,6	0,6	0,9	0,4	0,7	0,4	0,4	0,5	0,6	0,4	0,5

Çizelge 5.24 : Çalışanların toplam performans skorları.

	Toplam Performans Skoru
Çalışan 6	0,890
Çalışan 5	0,889
Çalışan 4	0,852
Çalışan 7	0,825
Çalışan 2	0,817
Çalışan 3	0,711
Çalışan 1	0,659
Çalışan 10	0,534
Çalışan 8	0,432
Çalışan 9	0,431

5.5 Uygulama Sonucu

Bu uygulamada, telekomünikasyon sektöründe faaliyet gösteren uluslararası bir firmanın Ar-Ge departmanındaki bir yazılım test ekibinin performansını değerlendirmek için uzmanlardan kullanılan 12 alt kriterin ağırlıklarını belirlemek üzere, önce 4 ana kriterin daha sonra da alt kriterlerin kendi grupları içinde ikili kıyaslamalarını yapmaları istenmiş ve Tamamlanmamış Aralık Değerli Sezgisel Tercih İlişkileri ile eksik karşılaştırmalar tamamlanmıştır. Daha sonra, hesaplanan bulanık ağırlıklar netleştirilmiş ve her bir alt kriterin ağırlığı, hiyerarşik olarak üst seviyesindeki kriterin ağırlığına göre elde edilmiştir.

Elde edilen sonuçlar, K1- Teknik beceriler ana kriterinin en yüksek öneme, K4- Ekibe katkı kriterinin ise en düşük öneme sahip olduğunu göstermiştir. Alt kriterler arasında ise, K12- Çıktı kalitesi ile K11-Teknik yeterlilik ve sorun çözme kriterleri yüksek

öneme sahipken; K31- Dürüstlük ve görevlerin tamamlanma süresi, K41- Ekip çalışmasına katkı, K22- Kendini adama, K21- Genel davranış, K13- Kendini geliştirme ve K33- Zorlu görevlerin başarımı orta derecede önemli kriterler;, K43- Yönlendiricilik ve sorumluluk alma, K32- Çoklu görevleri başarabilme, K42 Bilgi paylaşımı ve K23- Stres yönetimi nispeten daha düşük öneme sahip kriterler olarak görülmüştür.

Önceden performans skoru hesaplanırken her bir kriter için gözlemlenen skor eşit öneme sahipken, bulanık mantığın yeni uzantılarıyla hesaplanan ağırlıklar sayesinde, yazılım test ekibi çalışanlarının gösterdikleri performans gerçeği daha doğru yansıtır bir şekilde hesaplanabilir hale gelmiştir.

Daha sonra elde edilen toplam performans skoru sayesinde de çalışanlar arasında bir sıralama yapmak mümkün olmuştur.

6. SONUÇ VE ÖNERİLER

Şirketler arası rekabetin giderek arttığı ve nitelikli çalışan istihdam etme yarışlarının yaşandığı bilişim sektöründe insan kaynakları yönetiminin bir alt kategorisi olan performans yönetiminin önemi yadsınamaz bir gerçektir. Performansının yöneticileri tarafından doğru gözlemlenemediğini veya kendisini geliştirmek için doğru hedefler konmadığını düşünen çalışanlardaki motivasyon kaybı, çalışanların işten ayrılma sebeplerindendir. Bu sebeple, yöneticilerin kullandıkları performans ölçüm modelini çalışılan sektörün ve yapılan işin gereksinimleriyle paralel olacak şekilde uyarlamaları gerekmektedir.

Bu çalışmada da, yazılım sektöründe faaliyet gösteren ve yeni yazılım geliştirme ve proje yönetimi yaklaşımlarını uygulayan bir departmanın, performans değerlendirmesi problemini çözmek için öncelikle bulanık yaklaşımla TADSTB'ler kullanılarak kriter ağırlıkları belirlenmiş; daha sonra PROMETHEE metodundan türetilen performans gösterge fonksiyonlarıyla toplam performans skoru hesaplanmıştır. Scrum ve Devops yaklaşımlarda önemli olan çeviklik, takım olarak kendi kendine yetebilme, kalite odaklı olma gibi özellikleri sağlayabilmek için çalışanların da çapraz fonksiyonlu çalışabilecek teknik yeterliliğe sahip, takım çalışmasına yatkın, sorumluluk alabilen, iletişim yeteneği gelişmiş ve gelişime açık olması beklenmektedir. Performans değerlendirme kriterleri de bu beklentiler doğrultusunda belirlenmiştir.

Kullanılan kriterlerin dilsel terimler yardımıyla hesaplanan ağırlıkları, gözlemlenen performansın doğru bir şekilde değerlendirilmesini sağladığı gibi, bir sonraki dönem için çalışanların bireysel performans hedefleri belirlenip, kendilerini geliştirmeleri gereken noktalar tartışılırken, kendileri ve takım için hangi alanda kendilerini geliştirmenin daha faydalı olacağını da gözler önüne sermektedir.

Ek olarak, ağırlıkların hesaplamasını daha hızlı ve kolay yapabilmek için geliştirilen yazılım sayesinde yeni bir projeye ya da yazılım testi için geliştirilen yeni bir yaklaşıma göre, performans değerlendirmesi sırasında kullanılan kriterleri güncellemek gerektiğinde, yeni kriterlerin ağırlıklarını hesaplamak için uzmanlardan

bir kriterin diğerlerine göre önemini dilsel ifadelerle kıyaslamalarını istemek yeterli olacaktır. Bu noktada, uzmanlardan tek bir kriter için alınan ikili karşılaştırmaların doğru olduğu varsayılmaktadır. Yazılım, uzman sayısına göre değişen SAB vektörünü ve kriter sayısına göre değişen diğer tüm hesaplamaları girdi olarak verilen Excel dosyası sayesinde otomatik olarak yapacaktır. İleride yazılıma bir önyüz eklenerek, uzmanların karşılaştırmaları Excel yerine arayüzden yapıp, sonuçları da yine aynı arayüzde görmeleri sağlanabilir. Ayrıca yazılım, her bir kriter için uygun olan performans gösterge fonksiyonunu seçme olanağını uzmanlara sağlayarak, çalışanların toplam performans skorunu da hesaplayacak şekilde geliştirilebilir.

Gelecek çalışmalarda, performans değerlendirme problemi, tip-2 bulanık kümeler ve tereddütlü (hesitant) bulanık kümeler gibi yeni bulanık küme uzantılarıyla da ele alınabilir.

KAYNAKLAR

Ahmed, F. & Kilic, K. (2018). Fuzzy Analytic Hierarchy Process: A performance analysis of various algorithms. *Fuzzy Sets and Systems*. http://doi.org/10.1016/j.fss.2018.08.009

Ahmed, I., Sultana, I., Paul, S. K. & Azeem, A. (2013). Employee performance evaluation: A fuzzy approach. *International Journal of Productivity and Performance Management*. http://doi.org/10.1108/IJPPM-01-2013-0013

Ahn, J. Y., Han, K. S., Oh, S. Y. & Lee, C. D. (2011). An application of interval-valued intuitionistic fuzzy sets for medical diagnosis of headache. *International Journal of Innovative Computing, Information and Control*.

Alahyari, H., Horkoff, J., Matsson, O. & Egenvall, K. (2012). What Do Agile Teams Find Important for Their Success? *Proceedings - 25th Asia-Pacific Software Engineering Conference (APSEC), Nara, Japan, 2018.* pp. 474-483.https://doi.org/10.1109/APSEC.2018.00062

Al-Heyasi, A. (2018). Individuals Performance Measurement In Agile Software Development. *Eurasian Journal of Social Sciences*. http://doi.org/10.15604/ejss.2018.06.01.001

Alnaji, L. & Salameh, H. (2015). Performance-Measurement Framework to Evaluate Software Engineers for Agile Software-Development Methodology. *European Journal of Business and Management*.

Arbaiy, N. & Suradi, Z. (2007). Staff performance appraisal using fuzzy evaluation. In *IFIP International Federation for Information Processing*. http://doi.org/10.1007/978-0-387-74161-1_21

Armstrong, M. (2000). *Performance management: Key Strategies and Practical Guidelines* (2nd ed.). London: Kogan Page.

Armstrong, M. & Baron, A. (1998). *Performance management : the new realities*. London: Institute of Personnel and development.

Barki, H., Rivard, S. & Talbot, J. (2001). An integrative contingency model of software project risk management. *Journal of Management Information Systems.* https://doi.org/10.1080/07421222.2001.11045666

Bozbura, F. T., Beskese, A. & Kahraman, C. (2007). Prioritization of human capital measurement indicators using fuzzy AHP. *Expert Systems with Applications*. https://doi.org/10.1016/j.eswa.2006.02.006

Brans, J. P. & Vincke, P. (1985). A preference ranking organization method: the PROMETHEE method for MCDM. *Management Science*.

Chandran, S & Kandasamy, G. (2011). Fuzzy Performance Appraisal System. *Journal of Artificial Intelligent Systems and Machine Learning*. 3. 652-657.

Chen, T. (2008). "The Application of the Function Point Analysis in Software Developers' Performance Evaluation," *2008 4th International Conference on Wireless Communications, Networking and Mobile Computing*, Dalian, pp. 1-4. doi: 10.1109/WiCom.2008.1722

Chen, J. F., Hsieh, H. N. & Do, Q. H. (2015). Evaluating teaching performance based on fuzzy AHP and comprehensive evaluation approach. *Applied Soft Computing Journal*. https://doi.org/10.1016/j.asoc.2014.11.050

Chen, T. Y., Wang, H. P. & Lu, Y. Y. (2011). A multicriteria group decision-making approach based on interval-valued intuitionistic fuzzy sets: A comparative perspective. *Expert Systems with Applications*. http://doi.org/10.1016/j.eswa.2010.12.096

Chou, Y. C., Yen, H. Y., Sun, C. C. & Hon, J. S. (2014). Comparison of AHP and fuzzy AHP methods for human resources in science technology (HRST) performance index selection. In *IEEE International Conference on Industrial Engineering and Engineering Management*. http://doi.org/10.1109/IEEM.2013.6962520

Çelikyılmaz, A. ve Türksen, I. B. (2009), *Modeling Uncertainty with Fuzzy Logic: With Recent Theory and Applications*, Studies in Fuzziness and Soft Computing, Berlin: Springer-Verlag.

Gamble, R. F. & Hale, M. L. (2013). Assessing individual performance in Agile undergraduate software engineering teams. In *Proceedings - Frontiers in Education Conference, FIE*. http://doi.org/10.1109/FIE.2013.6685123

Gefen, D., Ridings, C. M. (2002). Implementation Team Responsiveness and User Evaluation of Customer Relationship Management: A Quasi-Experimental Design Study of Social Exchange Theory. *Journal of Management Information Systems, 19(1)*, 47–69. https://doi.org/10.1080/07421222.2002.11045717

Giannopoulos, A. (2015). Performance Management as a Process of Promoting Innovation in Software Industry. *Procedia - Social and Behavioral Sciences*. https://doi.org/10.1016/j.sbspro.2015.01.1216

Gök, A.C. (2015). Performans Değerlendirmesinde Bulanık Çok Kriterli Karar Verme Yaklaşımı: Türk İmalat İşletmeleri Örneği. (Doktora tezi). Karadeniz Teknik Üniversitesi, Sosyal Bilimler Enstitüsü, Trabzon

Gu, X. & Zhu, Q. (2006). Fuzzy multi-attribute decision-making method based on eigenvector of fuzzy attribute evaluation space. *Decision Support Systems.* https://doi.org/10.1016/j.dss.2004.08.001

Hüttermann M. (2012) Beginning DevOps for Developers. In: *DevOps for Developers*. Apress, Berkeley, CA

Kanij, T., Grundy, J. & Merkel, R. (2014). Performance appraisal of software testers. *Information and Software Technology*. http://doi.org/10.1016/j.infsof.2013.11.002

Kanij, T., Merkel, R. & Grundy, J. (2012). Performance assessment metrics for software testers. In *2012 5th International Workshop on Co-operative and Human Aspects of Software Engineering, CHASE 2012 - Proceedings* (pp. 63–65). http://doi.org/10.1109/CHASE.2012.6223025

Krishna Kaiser, A. (2018). Introduction to DevOps. In *Reinventing ITIL® in the Age of DevOps: Innovative Techniques to Make Processes Agile and Relevant* (pp. 1–35). https://doi.org/10.1007/978-1-4842-3976-6_1

Kusumawardani, R. P. & Agintiara, M. (2015). Application of Fuzzy AHP-TOPSIS Method for Decision Making in Human Resource Manager Selection Process. *Procedia Computer Science*. https://doi.org/10.1016/j.procs.2015.12.173

Lee, Keon Myung. (2004). Comparison of Interval-valued fuzzy sets, Intuitionistic fuzzy sets, and bipolar-valued fuzzy sets. Journal of Korean Institute of Intelligent Systems. 14. 10.5391/JKIIS.2004.14.2.125.

Li, D. F. (2011). Extension principles for interval-valued intuitionistic fuzzy sets and algebraic operations. *Fuzzy Optimization and Decision Making*. http://doi.org/10.1007/s10700-010-9095-9

Memik, B.F. (2017). Personel Performans Değerlendirme Süreci İçin Bulanık Ortamda Bütünleşik Bir Model Önerisi. (Yüksek lisans tezi). İstanbul Ticaret Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul.

Narayan, M. (1996). Evaluating Software Teams Effectively. *IFAC Proceedings Volumes*, *29*(2), 23–26. http://doi.org/10.1016/S1474-6670(17)43772-4

Öztayşi, B., Onar, S. C., Goztepe, K. & Kahraman, C. (2015). Evaluation of research proposals for grant funding using interval-valued intuitionistic fuzzy sets. *Soft Computing*. http://doi.org/10.1007/s00500-015-1853-8

Öztayşi, B., Onar, S. Ç., Kahraman, C. & Gök, M. (2016). *Performance Measurement Of Process Based Enterprise Applications Using Incomplete Interval-Valued Intuitionistic Preference Relations: The Case Of Call Centers*. https://doi.org/10.1142/9789813146976_0132

Parizi, R. M., Spoletini, P. & Singh, A. (2019). Measuring Team Members' Contributions in Software Engineering Projects using Git-driven Technology. *Proceedings - Frontiers in Education Conference, FIE*. https://doi.org/10.1109/FIE.2018.8658983

Podjavo, I.. & Berzisa, S. (2017). PERFORMANCE EVALUATION OF SOFTWARE DEVELOPMENT PROJECT TEAM. *Environment. Technology. Resources. Proceedings of the International Scientific and Practical Conference*. http://doi.org/10.17770/etr2017vol2.2543

Raza, M., Faria, J. P. & Salazar, R. (2017). Helping software engineering students analyzing their performance data: Tool support in an educational environment. In *Proceedings - 2017 IEEE/ACM 39th International Conference on Software Engineering Companion, ICSE-C 2017*. http://doi.org/10.1109/ICSE-C.2017.61

Rose, K. H. (2013). A Guide to the Project Management Body of Knowledge (PMBOK® Guide)-Fifth Edition. *Project Management Journal*. http://doi.org/10.1002/pmj.21345

Ross, T. J. (2010), *Fuzzy Logic with Engineering Applications*, Third Edition, United Kingdom: John Wiley and Sons.

Schwaber, K. & Sutherland, J. (2017). Scrum Guide | Scrum Guides. Retrieved from https://scrumguides.org/scrum-guide.html

Shaout, A. & Khalid Yousif, M. (2014). Employee Performance Appraisal System Using Fuzzy Logic. *International Journal of Computer Science and Information Technology*. http://doi.org/10.5121/ijcsit.2014.6401

Sheu, J. B. (2004). A hybrid fuzzy-based approach for identifying global logistics strategies. *Transportation Research Part E: Logistics and Transportation Review*. https://doi.org/10.1016/j.tre.2003.08.002

Sugeno, M. & Kang, G. T. (1986). Fuzzy modelling and control of multilayer incinerator. *Fuzzy Sets and Systems*, 18(3), 329–345. https://doi.org/10.1016/0165-0114(86)90010-2

Taticchi, P. (2010). *Business performance measurement and management: New contexts, themes and challenges*. *Business Performance Measurement and Management: New Contexts, Themes and Challenges*. http://doi.org/10.1007/978-3-642-04800-5

Trost, A. (2017). *The End of Performance Appraisal*. https://doi.org/10.1007/978-3-319-54235-5

Uyargil, C. & Hiperlink (Firm). (2008). İşletmelerde performans yönetimi sistemi : performansın planlanması değerlendirilmesi ve geliştirilmesi. İstanbul: Hiperlink. Retrieved from http://0-search.ebscohost.com.divit.library.itu.edu.tr/login.aspx?direct=true&db=nlebk&AN=659955&lang=tr&site=eds-live

West, D., Groll, J. (2007). *The convergence of Scrum and DevOps* [White paper]. <*https://scrumorg-website-prod.s3.amazonaws.com/drupal/2017-09/The%20Convergence%20of%20Scrum%20and%20DevOps.pdf*>, erişim tarihi: 13.02.2019

Xu, Z. (2007). Intuitionistic preference relations and their application in group decision making. *Information Sciences*. https://doi.org/10.1016/j.ins.2006.12.019

Xu, Z. S. & Chen, J. (2007). Approach to group decision making based on intervalvalued intuitionistic judgment matrices. *Systems Engineering – Theory and Practice*, 27(4), 126–133.

Yang, C. L., Huang, R. H. & Ho, J. J. (2009). Designing a multi criteria evaluation model for project teams. In *IEEM 2009 - IEEE International Conference on Industrial Engineering and Engineering Management*. http://doi.org/10.1109/IEEM.2009.5373258

Yousif, M. K. & Shaout, A. (2018). Fuzzy logic computational model for performance evaluation of Sudanese Universities and academic staff. *Journal of King Saud University - Computer and Information Sciences*. https://doi.org/10.1016/j.jksuci.2016.08.002

Zadeh, L. A. (1984). Making Computers Think Like People THINK LIKE PEOPLE. *IEEE Spectrum*, *21*(8), 26–32. http://doi.org/10.1109/MSPEC.1984.6370431

Zhang, C., Li, W. & Wang, L. (2011). AHP under the Intuitionistic Fuzzy environment. In *Proceedings - 2011 8th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2011*. http://doi.org/10.1109/FSKD.2011.6019593

Url-1 <*https://scrumorg-website-prod.s3.amazonaws.com/drupal/2017-09/The%20Convergence%20of%20Scrum%20and%20DevOps.pdf* >, erişim tarihi: 13.02.2019

Url-2 <*http://www.masters-consulting.de/fileadmin/web/pdf/2017/Agile_Service_Management_Guide_V1_031715.pdf*>, erişim tarihi: 13.02.2019

U**rl-3** <*https://agilemanifesto.org/principles.html*>, erişim tarihi: 13.02.2019

U**rl-4** < *https://www.agilealliance.org/agile101*/>, erişim tarihi: 14.02.2019

U**rl-5** <*https://www.visual-paradigm.com/scrum/what-is-agile-software-development/*>, erişim tarihi: 01.05.2019

U**rl-6** < *https://www.acmagile.com/devops-nedir/*>, erişim tarihi: 14.02.2019

EKLER

EK A1: App.java sınıfı
EK A2: IVIFN.java sınıfı
EK A3: LinguisticTerms.java sınıfı
EK A4: Experts.java sınıfı
EK A5: Equations.java sınıfı
EK A6: SortedPreferenceRelation.java sınıfı
EK B: Alt kriterlerin ağırlık hesaplama çizelgeleri

EK A1

```
package com.tezcan;

import java.io.File;
import java.io.FileInputStream;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

import org.apache.poi.hssf.usermodel.HSSFCell;
import org.apache.poi.hssf.usermodel.HSSFRow;
import org.apache.poi.hssf.usermodel.HSSFSheet;
import org.apache.poi.hssf.usermodel.HSSFWorkbook;
import org.apache.poi.poifs.filesystem.POIFSFileSystem;

import static com.tezcan.Equations.*;


public class App {

   public static void main(String[] args) {

         Expert experts[] = null;
         try {
                File file = new File("input/CCN_Test_criterias.xls");
            POIFSFileSystem fileSystem = new POIFSFileSystem(new
FileInputStream(file));
            HSSFWorkbook workBook = new HSSFWorkbook(fileSystem);
            HSSFSheet sheet = workBook.getSheetAt(0);
            HSSFRow row;
            HSSFCell cell;

            int rows; // Count of rows
            rows = sheet.getPhysicalNumberOfRows();

            int cols = 0; // Count of columns
            int tmp = 0;

            // The following part ensures that the data has been got properly even if it
doesn't start from first row
            for(int i = 0; i < 10 || i < rows; i++) {
               row = sheet.getRow(i);
               if(row != null) {
                  tmp = sheet.getRow(i).getPhysicalNumberOfCells();
                  if(tmp > cols) cols = tmp;
               }
            }

            experts = new Expert[rows-1];
```

```
            for(int r = 0; r < rows; r++) {
               row = sheet.getRow(r+1);
               if(row != null) {
                  experts[r] = new Expert();
                  for(int c = 0; c < cols; c++) {
                     cell = row.getCell((short)(c+1));
                     if(cell != null) {

         experts[r].getComparison().add(LinguisticTerms.convertToIvifn(cell.getStrin
gCellValue()));
                     }
                  }
               }
            }
         } catch(Exception ioe) {
            ioe.printStackTrace();
         }


      List<List<Ivifn>> matrix;
      for (int i = 0; i < experts.length; i++) {
         matrix = experts[i].getMatrix();
         System.out.println("Size:" + matrix.size());
         ArrayList<Ivifn> rowOne = new ArrayList<Ivifn>();
         for (int j = 0; j < experts[i].getComparison().size(); j++) {
            rowOne.add(experts[i].getComparison().get(j));
         }
         matrix.add(rowOne);

         for (int k = 1; k < rowOne.size(); k++) {
            ArrayList<Ivifn> otherRow = new ArrayList<Ivifn>();
            Ivifn reverse = new Ivifn();
            for (int j = 0; j < experts[i].getComparison().size(); j++) {
               if (j == 0) {
                  reverse = reverse(rowOne.get(k));
                  otherRow.add(reverse);
               } else if (j == k) {
                  otherRow.add(LinguisticTerms.ee);
               } else {
                  otherRow.add(calculateUnknownIvifn(reverse, rowOne.get(j)));
               }
            }
            matrix.add(otherRow);
         }

         for (int k = 0; k < matrix.size(); k++) {
            experts[i].getAggregated().add(aggregate(matrix.get(k)));
         }

         System.out.println("Expert " + (i + 1) + "'s pairwise comparison matrix:");
```

```
            System.out.println(experts[i].toString());
        }

        double[] owas = calculateOwa(experts.length);
        Ivifn ivifsWeight[] = new Ivifn[experts[0].getComparison().size()];

        for (int i = 0; i < experts[0].getComparison().size(); i++) {
            List<SortedPreferenceRelation> list = new
ArrayList<SortedPreferenceRelation>();
            for (Expert e : experts) {
                SortedPreferenceRelation sorted = new SortedPreferenceRelation();
                sorted.setIvifn(e.getAggregated().get(i));
                sorted.setScore(calculateScore(sorted.getIvifn()));
                list.add(sorted);
            }

            Collections.sort(list);

            // Step 6 and 7

            for (SortedPreferenceRelation sortedPreferenceRelation : list) {
                System.out.println("sortedPreferenceRelation: " +
sortedPreferenceRelation.getIvifn().toString() + " Score: " +
sortedPreferenceRelation.getScore());
            }

            ivifsWeight[i] = calculateIvifsWeifght(list, owas);
        }


        double defuzzifiedWeight[] = new double[experts[0].getComparison().size()];
        //   double defuzzifiedWeight[] = {0.579f, 0.451f, 0.378f, 0.403f, 0.354f,
0.359f};

        for (int i = 0; i < experts[0].getComparison().size(); i++) {
            defuzzifiedWeight[i] = defuzzify(ivifsWeight[i]);
        }

        double normalized[] = normalize(defuzzifiedWeight);
        System.out.println("Kriter No\tADSBS Ağırlıkları\t\t\t\tNetleştirilmiş
Ağırlıklar\t\tNormalize edilmiş");
        for (
                int i = 0;
                i < normalized.length; i++) {
            System.out.println("C" +(i+1) + "\t\t\t" + ivifsWeight[i] + "\t\t" +
String.format("%.3f", defuzzifiedWeight[i]) + "\t\t\t\t\t\t" + String.format("%.3f",
normalized[i]));
        }
    }
}
```

EK A2

```
package com.tezcan;

public class Ivifn {

   private double muLower;
   private double muUpper;
   private double vLower;
   private double vUpper;

   public Ivifn() {

   }

   public Ivifn(double muLower, double muUpper, double vLower, double vUpper) {
      this.setMuLower(muLower);
      this.setMuUpper(muUpper);
      this.setVLower(vLower);
      this.setVUpper(vUpper);
   }

   /**
    * @return the vUpper
    */
   public double getVUpper() {
      return vUpper;
   }

   /**
    * @param vUpper the vUpper to set
    */
   public void setVUpper(double vUpper) {
      this.vUpper = vUpper;
   }

   /**
    * @return the muLower
    */
   public double getMuLower() {
      return muLower;
   }

   /**
    * @param muLower the muLower to set
    */
   public void setMuLower(double muLower) {
      this.muLower = muLower;
   }
```

```
    /**
     * @return the muUpper
     */
    public double getMuUpper() {
        return muUpper;
    }

    /**
     * @param muUpper the muUpper to set
     */
    public void setMuUpper(double muUpper) {
        this.muUpper = muUpper;
    }

    /**
     * @return the vLower
     */
    public double getVLower() {
        return vLower;
    }

    /**
     * @param vLower the vLower to set
     */
    public void setVLower(double vLower) {
        this.vLower = vLower;
    }


    public String toString() {
        return "([" + formatDouble(getMuLower()) + ", " +
formatDouble(getMuUpper()) + "],[" +
                formatDouble(getVLower()) + ", " + formatDouble(getVUpper()) + "])";
    }

    private String formatDouble(double doubleValue) {
        return String.format("%.3f", doubleValue);
    }
}
```

EK A3

```
package com.tezcan;
public class LinguisticTerms {

   /* ee = Exactly Equal */
   public static Ivifn ee = new Ivifn(0.5f, 0.5f, 0.5f, 0.5f);
   /* al = Absolutely Low */
   public static Ivifn al = new Ivifn(0.1f, 0.25f, 0.65f, 0.75f);
   /* vl = Very Low */
   public static Ivifn vl = new Ivifn(0.15f, 0.3f, 0.6f, 0.7f);
   /* l = Low */
   public static Ivifn l = new Ivifn(0.2f, 0.35f, 0.55f, 0.65f);
   /* ml = Medium Low */
   public static Ivifn ml = new Ivifn(0.25f, 0.4f, 0.5f, 0.6f);
   /* ae = Approximately Equal */
   public static Ivifn ae = new Ivifn(0.45f, 0.55f, 0.3f, 0.45f);
   /* mh = Medium High */
   public static Ivifn mh = new Ivifn(0.5f, 0.6f, 0.25f, 0.4f);
   /* h = High */
   public static Ivifn h = new Ivifn(0.55f, 0.65f, 0.2f, 0.35f);
   /* vh = Very High */
   public static Ivifn vh = new Ivifn(0.6f, 0.7f, 0.15f, 0.3f);
   /* ah = Absolutely High */
   public static Ivifn ah = new Ivifn(0.65f, 0.75f, 0.1f, 0.25f);

   public static Ivifn convertToIvifn(String term) {
                  if (term.equalsIgnoreCase("EE")) {
                           return ee;
                  } else if (term.equalsIgnoreCase("AL")) {
                           return al;
                  } else if (term.equalsIgnoreCase("VL")) {
                           return vl;
                  } else if (term.equalsIgnoreCase("L")) {
                           return l;
                  } else if (term.equalsIgnoreCase("ML")) {
                           return ml;
                  } else if (term.equalsIgnoreCase("AE")) {
                           return ae;
                  } else if (term.equalsIgnoreCase("MH")) {
                           return mh;
                  } else if (term.equalsIgnoreCase("H")) {
                           return h;
                  } else if (term.equalsIgnoreCase("VH")) {
                           return vh;
                  } else if (term.equalsIgnoreCase("AH")) {
                           return ah;
                  }

                  return ee;        } }
```

EK A4

```
package com.tezcan;

import java.util.ArrayList;
import java.util.List;

public class Expert {

   private List<Ivifn> comparison = new ArrayList<Ivifn>();

   private List<List<Ivifn>> matrix = new ArrayList<List<Ivifn>>();

   private List<Ivifn> aggregated = new ArrayList<Ivifn>();

   /**
    * @return the comparison
    */
   public List<Ivifn> getComparison() {
      return comparison;
   }

   /**
    * @param comparison the comparison to set
    */
   public void setComparison(List<Ivifn> comparison) {
      this.comparison = comparison;
   }

   public List<List<Ivifn>> getMatrix() {
      return matrix;
   }

   public void setMatrix(List<List<Ivifn>> matrix) {
      this.matrix = matrix;
   }

   public String toString() {
      String output = "";
      for (int i = 0; i < matrix.size(); i++) {
         int length = matrix.get(i).size();
         for (int j = 0; j < length; j++) {
            output += " - " + matrix.get(i).get(j).toString();
         }
         output += "\n";
      }

      output += "\n-------------------------------------------------------\n";
      output += "Aggregated preference relations of Expert \n\n";
```

```
        for (int i = 0; i < aggregated.size(); i++) {
            output += aggregated.get(i).toString() + "\n";
        }
        return output;
    }

    /**
     * @return the aggregated
     */
    public List<Ivifn> getAggregated() {
        return aggregated;
    }

    /**
     * @param aggregated the aggregated to set
     */
    public void setAggregated(List<Ivifn> aggregated) {
        this.aggregated = aggregated;
    }

}
```

EK A5

```
package com.tezcan;

import java.util.List;

import static java.lang.Math.round;

public class Equations {

   // For Step 2
   public static Ivifn reverse(Ivifn ivifn) {

      Ivifn reverse = new Ivifn(ivifn.getVLower(), ivifn.getVUpper(),
ivifn.getMuLower(), ivifn.getMuUpper());
      return reverse;
   }

   // For Step 2
   // Eq. (4.39)
   public static Ivifn calculateUnknownIvifn(Ivifn first, Ivifn second) {
      Ivifn ivifn = new Ivifn();

      ivifn.setMuLower((first.getMuLower() + second.getMuLower()) / 2);
      ivifn.setMuUpper((first.getMuUpper() + second.getMuUpper()) / 2);
      ivifn.setVLower((first.getVLower() + second.getVLower()) / 2);
      ivifn.setVUpper((first.getVUpper() + second.getVUpper()) / 2);

      return ivifn;
   }

   // For Step 3
   //Eq. (4.41)
   public static Ivifn aggregate(List<Ivifn> criteriaRow) {
      Ivifn ivifn = new Ivifn();

      double muLow = 1.0, muUp = 1.0, vLow = 1.0, vUp = 1.0;
      for (Ivifn i : criteriaRow) {
         muLow *= (1 - i.getMuLower());
         muUp *= (1 - i.getMuUpper());
         vLow *= i.getVLower();
         vUp *= i.getVUpper();
      }

      muLow = 1 - Math.pow(muLow, (double) 1 / criteriaRow.size());
      muUp = 1 - Math.pow(muUp, (double) 1 / criteriaRow.size());
      vLow = Math.pow(vLow, (double) 1 / criteriaRow.size());
      vUp = Math.pow(vUp, (double) 1 / criteriaRow.size());

      ivifn.setMuLower(muLow);
```

```
        ivifn.setMuUpper(muUp);
        ivifn.setVLower(vLow);
        ivifn.setVUpper(vUp);

        return ivifn;
    }

    // For Step 4
    // Eq. (4.43)
    public static double calculateScore(Ivifn ivifn) {

        double score = (ivifn.getMuLower() + ivifn.getMuUpper() - ivifn.getVLower()
- ivifn.getVUpper()) / 2;

        return score;
    }

    // For Step 5
    // Eq. (4.42)
    public static double[] calculateOwa(int dimension) {

        double[] owaVector = new double[dimension];
        double mu = (1 + ((double) dimension)) / 2;
        double sumForSigma = 0;
        double sumForVector = 0;

        for (int i = 1; i <= dimension; i++) {
            sumForSigma += Math.pow((i - mu), 2);
        }

        double sigma = Math.sqrt(sumForSigma / dimension);

        for (int i = 1; i <= dimension; i++) {
            sumForVector += Math.exp(-(Math.pow((i - mu), 2) / (2 * (Math.pow(sigma,
2)))));
        }

        for (int k = 0; k < dimension; k++) {
            owaVector[k] = (Math.exp(-(Math.pow((k + 1 - mu), Math.pow(2,
dimension)) / (2 * Math.pow(sigma, 2))))) / sumForVector;

             System.out.println("Weight " + k + " for dimension " + dimension + ": " +
owaVector[k]);
        }

        return owaVector;
    }

    // For Step 6
    // Eq. (4.41)
```

```
   public static Ivifn calculateIvifsWeifght(List<SortedPreferenceRelation>
criteriaColumn, double[] owaVector) {
      Ivifn ivifn = new Ivifn();
      double muLow = 1.0, muUp = 1.0, vLow = 1.0, vUp = 1.0;
      for (int j = 0; j < owaVector.length; j++) {
         Ivifn sortedIvifn = criteriaColumn.get(j).getIvifn();
         muLow *= Math.pow((1 - sortedIvifn.getMuLower()), owaVector[j]);
         muUp *= Math.pow((1 - sortedIvifn.getMuUpper()), owaVector[j]);
         vLow *= Math.pow(sortedIvifn.getVLower(), owaVector[j]);
         vUp *= Math.pow(sortedIvifn.getVUpper(), owaVector[j]);
      }

      muLow = 1 - muLow;
      muUp = 1 - muUp;

      ivifn.setMuLower(muLow);
      ivifn.setMuUpper(muUp);
      ivifn.setVLower(vLow);
      ivifn.setVUpper(vUp);

      return ivifn;
   }

   // For Step 7
   // Eq. (4.45)
   public static double defuzzify(Ivifn ivifn) {

      double defuzzified = (ivifn.getMuLower() + ivifn.getMuUpper() + (1 -
ivifn.getVLower()) + (1 - ivifn.getVUpper())
            + ivifn.getMuLower() * ivifn.getMuUpper()
            - Math.sqrt((1 - ivifn.getVLower()) * (1 - ivifn.getVUpper()))) / 4;

      return defuzzified;
   }

   // For Step 7
   public static double[] normalize(double[] array) {

      double sum = (double) 0;

      double normalized[] = new double[array.length];

      for (int i = 0; i < array.length; i++) {
         sum += array[i];
      }
      for (int i = 0; i < array.length; i++) {
         normalized[i] = array[i] / sum;
      }
      return normalized;
   }}
```

EK A6

```
package com.tezcan;

public class SortedPreferenceRelation implements
Comparable<SortedPreferenceRelation> {

   private Ivifn ivifn;
   private double score;

   /**
    * @return the score
    */
   public double getScore() {
      return score;
   }

   /**
    * @param score the score to set
    */
   public void setScore(double score) {
      this.score = score;
   }

   /**
    * @return the ivifn
    */
   public Ivifn getIvifn() {
      return ivifn;
   }

   /**
    * @param ivifn the ivifn to set
    */
   public void setIvifn(Ivifn ivifn) {
      this.ivifn = ivifn;
   }

   public int compareTo(SortedPreferenceRelation o) {
      return (int) (10000 * (o.getScore() - this.getScore()));
   }
}
```

EK B

Çizelge B.1 : K1'in alt kriterleri için Uzman 1'in tamamlanmış ikili karşılaştırma matrisi.

Uzman 1	K11	K12	K13
K11	([0,500, 0,500],[0,500, 0,500])	([0,200, 0,350],[0,550, 0,650])	([0,550, 0,650],[0,200, 0,350])
K12	([0,550, 0,650],[0,200, 0,350])	([0,500, 0,500],[0,500, 0,500])	**([0,550, 0,650],[0,200, 0,350])**
K13	([0,200, 0,350],[0,550, 0,650])	**([0,200, 0,350],[0,550, 0,650])**	([0,500, 0,500],[0,500, 0,500])

Çizelge B.2 : K1'in alt kriterleri için Uzman 2'nin tamamlanmış ikili karşılaştırma matrisi.

Uzman 2	K11	K12	K13
K11	([0,500, 0,500],[0,500, 0,500])	([0,250, 0,400],[0,500, 0,600])	([0,450, 0,550],[0,300, 0,450])
K12	([0,500, 0,600],[0,250, 0,400])	([0,500, 0,500],[0,500, 0,500])	**([0,475, 0,575],[0,275, 0,425])**
K13	([0,300, 0,450],[0,450, 0,550])	**([0,275, 0,425],[0,475, 0,575])**	([0,500, 0,500],[0,500, 0,500])

Çizelge B.3 : K1'in alt kriterleri için Uzman 3'ün tamamlanmış ikili karşılaştırma matrisi.

Uzman 3	K11	K12	K13
K11	([0,500, 0,500],[0,500, 0,500])	([0,250, 0,400],[0,500, 0,600])	([0,600, 0,700],[0,150, 0,300])
K12	([0,500, 0,600],[0,250, 0,400])	([0,500, 0,500],[0,500, 0,500])	**([0,550, 0,650],[0,200, 0,350])**
K13	([0,150, 0,300],[0,600, 0,700])	**([0,200, 0,350],[0,550, 0,650])**	([0,500, 0,500],[0,500, 0,500])

Çizelge B.4 : K2'nin alt kriterleri için Uzman 1'in tamamlanmış ikili karşılaştırma matrisi.

Uzman 1	K21	K22	K23
K21	([0,500, 0,500],[0,500, 0,500])	([0,250, 0,400],[0,500, 0,600])	([0,500, 0,600],[0,250, 0,400])
K22	([0,500, 0,600],[0,250, 0,400])	([0,500, 0,500],[0,500, 0,500])	**([0,500, 0,600],[0,250, 0,400])**
K23	([0,250, 0,400],[0,500, 0,600])	**([0,250, 0,400],[0,500, 0,600])**	([0,500, 0,500],[0,500, 0,500])

Çizelge B.5 : K2'nin alt kriterleri için Uzman 2'nin tamamlanmış ikili karşılaştırma matrisi.

Uzman 2	K21	K22	K23
K21	([0,500, 0,500],[0,500, 0,500])	([0,250, 0,400],[0,500, 0,600])	([0,500, 0,600],[0,250, 0,400])
K22	([0,500, 0,600],[0,250, 0,400])	([0,500, 0,500],[0,500, 0,500])	**([0,500, 0,600],[0,250, 0,400])**
K23	([0,250, 0,400],[0,500, 0,600])	**([0,250, 0,400],[0,500, 0,600])**	([0,500, 0,500],[0,500, 0,500])

Çizelge B.6 : K2'nin alt kriterleri için Uzman 3'ün tamamlanmış ikili karşılaştırma matrisi.

Uzman 3	K21	K22	K23
K21	([0,500, 0,500],[0,500, 0,500])	([0,500, 0,600],[0,250, 0,400])	([0,500, 0,600],[0,250, 0,400])
K22	([0,250, 0,400],[0,500, 0,600])	([0,500, 0,500],[0,500, 0,500])	**([0,375, 0,500],[0,375, 0,500])**
K23	([0,250, 0,400],[0,500, 0,600])	**([0,375, 0,500],[0,375, 0,500])**	([0,500, 0,500],[0,500, 0,500])

Çizelge B.7 : K3'ün alt kriterleri için Uzman 1'in tamamlanmış ikili karşılaştırma matrisi.

Uzman 1	K31	K32	K33
K31	([0,500, 0,500],[0,500, 0,500])	([0,500, 0,600],[0,250, 0,400])	([0,500, 0,600],[0,250, 0,400])
K32	([0,250, 0,400],[0,500, 0,600])	([0,500, 0,500],[0,500, 0,500])	**([0,375, 0,500],[0,375, 0,500])**
K33	([0,250, 0,400],[0,500, 0,600])	**([0,375, 0,500],[0,375, 0,500])**	([0,500, 0,500],[0,500, 0,500])

Çizelge B.8 : K3'ün alt kriterleri için Uzman 2'nin tamamlanmış ikili karşılaştırma matrisi.

Uzman 2	K31	K32	K33
K31	([0,500, 0,500],[0,500, 0,500])	([0,650, 0,750],[0,100, 0,250])	([0,600, 0,700],[0,150, 0,300])
K32	([0,100, 0,250],[0,650, 0,750])	([0,500, 0,500],[0,500, 0,500])	**([0,350, 0,475],[0,400, 0,525])**
K33	([0,150, 0,300],[0,600, 0,700])	**([0,400, 0,525],[0,350, 0,475])**	([0,500, 0,500],[0,500, 0,500])

Çizelge B.9 : K3'ün alt kriterleri için Uzman 3'ün tamamlanmış ikili karşılaştırma matrisi.

Uzman 3	K31	K32	K33
K31	([0,500, 0,500],[0,500, 0,500])	([0,500, 0,600],[0,250, 0,400])	([0,250, 0,400],[0,500, 0,600])
K32	([0,250, 0,400],[0,500, 0,600])	([0,500, 0,500],[0,500, 0,500])	**([0,250, 0,400],[0,500, 0,600])**
K33	([0,500, 0,600],[0,250, 0,400])	**([0,500, 0,600],[0,250, 0,400])**	([0,500, 0,500],[0,500, 0,500])

Çizelge B.10 : K4'ün alt kriterleri için Uzman 1'in tamamlanmış ikili karşılaştırma matrisi.

Uzman 1	K41	K42	K43
K41	([0,500, 0,500],[0,500, 0,500])	([0,600, 0,700],[0,150, 0,300])	([0,500, 0,600],[0,250, 0,400])
K42	([0,150, 0,300],[0,600, 0,700])	([0,500, 0,500],[0,500, 0,500])	**([0,325, 0,450],[0,425, 0,550])**
K43	([0,250, 0,400],[0,500, 0,600])	**([0,425, 0,550],[0,325, 0,450])**	([0,500, 0,500],[0,500, 0,500])

Çizelge B.11 : K4'ün alt kriterleri için Uzman 2'nin tamamlanmış ikili karşılaştırma matrisi.

Uzman 2	K41	K42	K43
K41	([0,500, 0,500],[0,500, 0,500])	([0,450, 0,550],[0,300, 0,450])	([0,500, 0,600],[0,250, 0,400])
K42	([0,300, 0,450],[0,450, 0,550])	([0,500, 0,500],[0,500, 0,500])	**([0,400, 0,525],[0,350, 0,475])**
K43	([0,250, 0,400],[0,500, 0,600])	**([0,350, 0,475],[0,400, 0,525])**	([0,500, 0,500],[0,500, 0,500]

Çizelge B.12 : K4'ün alt kriterleri için Uzman 3'ün tamamlanmış ikili karşılaştırma matrisi.

Uzman 3	K41	K42	K43
K41	([0,500, 0,500],[0,500, 0,500])	([0,500, 0,600],[0,250, 0,400])	([0,500, 0,600],[0,250, 0,400])
K42	([0,250, 0,400],[0,500, 0,600])	([0,500, 0,500],[0,500, 0,500])	**([0,375, 0,500],[0,375, 0,500])**
K43	([0,250, 0,400],[0,500, 0,600])	**([0,375, 0,500],[0,375, 0,500])**	([0,500, 0,500],[0,500, 0,500])

Çizelge B.13 : K1'in alt kriterleri için uzmanların birleştirilmiş tercih değerleri.

	Uzman 1	Uzman 2	Uzman 3
K11	([0,435, 0,515],[0,380, 0,485])	([0,409, 0,487],[0,422, 0,513])	([0,469, 0,552],[0,335, 0,448])
K12	([0,534, 0,606],[0,271, 0,394])	([0,492, 0,560],[0,325, 0,440])	([0,517, 0,588],[0,292, 0,412])
K13	([0,316, 0,404],[0,533, 0,596])	([0,367, 0,459],[0,475, 0,541])	([0,302, 0,390],[0,548, 0,610])

Çizelge B.14 : K2'nin alt kriterleri için uzmanların birleştirilmiş tercih değerleri.

	Uzman 1	Uzman 2	Uzman 3
K21	([0,428, 0,507],[0,397, 0,493])	([0,428, 0,507],[0,397, 0,493])	([0,500, 0,569],[0,315, 0,431])
K22	([0,500, 0,569],[0,315, 0,431])	([0,500, 0,569],[0,315, 0,431])	([0,383, 0,469],[0,454, 0,531])
K23	([0,345, 0,435],[0,500, 0,565])	([0,345, 0,435],[0,500, 0,565])	([0,383, 0,469],[0,454, 0,531])

Çizelge B.15 : K3'ün alt kriterleri için uzmanların birleştirilmiş tercih değerleri.

	Uzman 1	Uzman 2	Uzman 3
K31	([0,500, 0,569],[0,315, 0,431])	([0,588, 0,665],[0,196, 0,335])	([0,428, 0,507],[0,397, 0,493])
K32	([0,383, 0,469],[0,454, 0,531])	([0,336, 0,418],[0,507, 0,582])	([0,345, 0,435],[0,500, 0,565])
K33	([0,383, 0,469],[0,454, 0,531])	([0,366, 0,450],[0,472, 0,550])	([0,500, 0,569],[0,315, 0,431])

Çizelge B.16 : K4'ün alt kriterleri için uzmanların birleştirilmiş tercih değerleri.

	Uzman 1	Uzman 2	Uzman 3
K41	([0,536, 0,609],[0,266, 0,391])	([0,484, 0,552],[0,335, 0,448])	([0,500, 0,569],[0,315, 0,431])
K42	([0,340, 0,423],[0,503, 0,577])	([0,406, 0,493],[0,429, 0,507])	([0,383, 0,469],[0,454, 0,531])
K43	([0,400, 0,487],[0,433, 0,513])	([0,375, 0,460],[0,464, 0,540])	([0,383, 0,469],[0,454, 0,531])

Çizelge B.17 : K1'in alt kriterleri için uzmanların birleştirilmiş tercih değerlerinin skorları

	Birleştirilmiş Değer	Skor
Uzman 1	([0,435, 0,515],[0,380, 0,485])	0,043
Uzman 2	([0,409, 0,487],[0,422, 0,513])	-0,019
Uzman 3	([0,469, 0,552],[0,335, 0,448])	0,1188
Uzman 1	([0,534, 0,606],[0,271, 0,394])	0,237
Uzman 2	([0,492, 0,560],[0,325, 0,440])	0,1436
Uzman 3	([0,517, 0,588],[0,292, 0,412])	0,2003
Uzman 1	([0,316, 0,404],[0,533, 0,596])	-0,204
Uzman 2	([0,367, 0,459],[0,475, 0,541])	-0,095
Uzman 3	([0,302, 0,390],[0,548, 0,610])	-0,234

Çizelge B.18 : K2'nin alt kriterleri için uzmanların birleştirilmiş tercih değerlerinin skorları

	Birleştirilmiş Değer	Skor
Uzman 1	([0,428, 0,507],[0,397, 0,493])	0,022
Uzman 2	([0,428, 0,507],[0,397, 0,493])	0,022
Uzman 3	([0,500, 0,569],[0,315, 0,431])	0,162
Uzman 1	([0,500, 0,569],[0,315, 0,431])	0,1616
Uzman 2	([0,500, 0,569],[0,315, 0,431])	0,1616
Uzman 3	([0,383, 0,469],[0,454, 0,531])	-0,0667
Uzman 1	([0,345, 0,435],[0,500, 0,565])	-0,142
Uzman 2	([0,345, 0,435],[0,500, 0,565])	-0,142
Uzman 3	([0,383, 0,469],[0,454, 0,531])	-0,067

Çizelge B.19 : K3'ün alt kriterleri için uzmanların birleştirilmiş tercih değerlerinin skorları

	Birleştirilmiş Değer	Skor
Uzman 1	([0,500, 0,569],[0,315, 0,431])	0,162
Uzman 2	([0,588, 0,665],[0,196, 0,335])	0,361
Uzman 3	([0,428, 0,507],[0,397, 0,493])	0,022
Uzman 1	([0,383, 0,469],[0,454, 0,531])	-0,0667
Uzman 2	([0,336, 0,418],[0,507, 0,582])	-0,1669
Uzman 3	([0,345, 0,435],[0,500, 0,565])	-0,1422
Uzman 1	([0,383, 0,469],[0,454, 0,531])	-0,067
Uzman 2	([0,366, 0,450],[0,472, 0,550])	-0,103
Uzman 3	([0,500, 0,569],[0,315, 0,431])	0,162

Çizelge B.20 : K4'ün alt kriterleri için uzmanların birleştirilmiş tercih değerlerinin skorları

	Birleştirilmiş Değer	Skor
Uzman 1	([0,536, 0,609],[0,266, 0,391])	0,244
Uzman 2	([0,484, 0,552],[0,335, 0,448])	0,126
Uzman 3	([0,500, 0,569],[0,315, 0,431])	0,162
Uzman 1	([0,340, 0,423],[0,503, 0,577])	-0,1588
Uzman 2	([0,406, 0,493],[0,429, 0,507])	-0,0189
Uzman 3	([0,383, 0,469],[0,454, 0,531])	-0,0667
Uzman 1	([0,400, 0,487],[0,433, 0,513])	-0,029
Uzman 2	([0,375, 0,460],[0,464, 0,540])	-0,084
Uzman 3	([0,383, 0,469],[0,454, 0,531])	-0,067

Çizelge B.21 : K1'in alt kriterlerinin skorlara göre sıralanmış birleştirilmiş değerleri.

	1	2	3
K11	([0,469, 0,552],[0,335, 0,448])	([0,435, 0,515],[0,380, 0,485])	([0,409, 0,487],[0,422, 0,513])
K12	([0,534, 0,606],[0,271, 0,394])	([0,517, 0,588],[0,292, 0,412])	([0,492, 0,560],[0,325, 0,440])
K13	([0,367, 0,459],[0,475, 0,541])	([0,316, 0,404],[0,533, 0,596])	([0,302, 0,390],[0,548, 0,610])

Çizelge B.22 : K2'nin alt kriterlerinin skorlara göre sıralanmış birleştirilmiş değerleri.

	1	2	3
K21	([0,500, 0,569],[0,315, 0,431])	([0,428, 0,507],[0,397, 0,493])	([0,428, 0,507],[0,397, 0,493])
K22	([0,500, 0,569],[0,315, 0,431])	([0,500, 0,569],[0,315, 0,431])	([0,383, 0,469],[0,454, 0,531])
K23	([0,383, 0,469],[0,454, 0,531])	([0,345, 0,435],[0,500, 0,565])	([0,345, 0,435],[0,500, 0,565])

Çizelge B.23 : K3'ün alt kriterlerinin skorlara göre sıralanmış birleştirilmiş değerleri.

K31	([0,588, 0,665],[0,196, 0,335])	([0,500, 0,569],[0,315, 0,431])	([0,428, 0,507],[0,397, 0,493])
K32	([0,383, 0,469],[0,454, 0,531])	([0,345, 0,435],[0,500, 0,565])	([0,336, 0,418],[0,507, 0,582])
K33	([0,500, 0,569],[0,315, 0,431])	([0,383, 0,469],[0,454, 0,531])	([0,366, 0,450],[0,472, 0,550])
K31	([0,588, 0,665],[0,196, 0,335])	([0,500, 0,569],[0,315, 0,431])	([0,428, 0,507],[0,397, 0,493])

Çizelge B.24 : K4'ün alt kriterlerinin skorlara göre sıralanmış birleştirilmiş değerleri.

K41	([0,536, 0,609],[0,266, 0,391])	([0,500, 0,569],[0,315, 0,431])	([0,484, 0,552],[0,335, 0,448])
K42	([0,406, 0,493],[0,429, 0,507])	([0,383, 0,469],[0,454, 0,531])	([0,340, 0,423],[0,503, 0,577])
K43	([0,400, 0,487],[0,433, 0,513])	([0,383, 0,469],[0,454, 0,531])	([0,375, 0,460],[0,464, 0,540])
K41	([0,536, 0,609],[0,266, 0,391])	([0,500, 0,569],[0,315, 0,431])	([0,484, 0,552],[0,335, 0,448])

ÖZGEÇMİŞ

Ad-Soyad **:** Işılay Tezcan

Doğum Tarihi ve Yeri **:** Tekirdag, 05.09.1987

E-posta **:** tezcani@itu.edu.tr

ÖĞRENİM DURUMU:

- **Lisans** **:** 2010, İstanbul Teknik Üniversitesi, Elektrik-Elektronik Fakültesi, Telekomünikasyon Mühendisliği

MESLEKİ DENEYİM VE ÖDÜLLER:

- 2012-2014 yılları arasında Alcatel Türkiye'de Yazılım Test Mühendisi olarak çalıştı.
- 2014 yılından itibaren Huawei Türkiye Ar-Ge Merkezinde Test Takım Lideri olarak çalışmaktadır.