NESNE YÖNELİMLİ YAZILIM DİLLERİNİN ANALİTİK HİYERARŞİ VE ANALİTİK NETWORK PROSESİ İLE KARŞILAŞTIRILMASI VE DEĞERLENDİRİLMESİ

Zelal ANIK

YÜKSEK LİSANS TEZİ
ENDÜSTRİ MÜHENDİSLİĞİ

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HAZİRAN 2007
ANKARA

Zelal ANIK tarafından hazırlanan NESNE YÖNELİMLİ YAZILIM DİLLERİNİN ANALİTİK HİYERARŞİ VE ANALİTİK NETWORK PROSESİ İLE KARŞILAŞTIRILMASI VE DEĞERLENDİRİLMESİ adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

Doç. Dr. Ö. Faruk BAYKOÇ
Tez Yöneticisi

Bu çalışma, jürimiz tarafından oy birliği ile Endüstri Mühendisliği Anabilim Dalında Yüksek lisans tezi olarak kabul edilmiştir.

Başkan: : Prof. Dr. Zülal GÜNGÖR

Üye : Doç. Dr. Ö. Faruk BAYKOÇ

Üye : Doç. Dr. İhsan ALP

Tarih : 06/06/2007

Bu tez, Gazi Üniversitesi Fen Bilimleri Enstitüsü tez yazım kurallarına uygundur.

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

Zelal ANIK

NESNE YÖNELİMLİ YAZILIM DİLLERİNİN ANALİTİK HİYERARŞİ VE ANALİTİK NETWORK PROSESİ İLE KARŞILAŞTIRILMASI VE DEĞERLENDİRİLMESİ
(Yüksek Lisans Tezi)

Zelal ANIK

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
Haziran 2007

ÖZET

Yazılım teknolojisindeki değişim 1970'lerde yapısal programlamanın yararlarının fark edilmesiyle başlamış, 1990'larda nesne yönelimli programlamanın iyice yaygınlaşmasıyla yazılım geliştirme süreci ivme kazanmıştır. Nesne yönelimli programlama daha anlaşılır, daha iyi organize olmuş ve incelenmesi, değiştirilmesi, hata ayıklaması daha kolay programlar üretmeye elverişlidir. Sağladığı birçok kolaylık nedeniyle günümüzde çok fazla talep görmektedir. Nesne yönelimli programlamanın önümüzdeki yıllarda programlama yöntemlerinde kilit nokta olacağı gerçeği bu çalışmanın tercih edilme sebebi olmuştur.

Bu çalışmada öncelikle iyi bir programlama dilinin sahip olması gereken özellikler arasındaki öncelikleri belirlemek, programlama dili seçimini kolaylaştıracak bir araç bulmak ve bu süreci çok daha düzgün hale getirmek hedeflenmiştir. Ancak uygun programlama dilini seçmek bir çok faktörü dikkate almayı gerektiren bir çeşit çok kriterli karar verme problemidir. Bu çalışmada nesne yönelimli yazılım dili seçimi problemi Analitik Hiyerarşi Prosesi(AHP) ve Analitik Network Prosesi(ANP) ile çözülmeye çalışılmış, elde edilen sonuçlar değerlendirilmiş, karşılaştırılmış ve bu süreç için uygunlukları araştırılmıştır. Literatürde nesne yönelimli yazılım dilleri için ANP ile yapılmış

herhangi bir çalışmaya rastlanmamış olup çalışma bu yönüyle de bir ilk olma özelliği taşımaktadır.

Bilim Kodu : 906.1.071
Anahtar Kelimeler : Nesne yönelimli programlama dilleri, Analitik Hiyerarşi Prosesi, Analitik Network Prosesi
Sayfa Adedi : 103
Tez Yöneticisi : Doç. Dr. Ö. Faruk BAYKOÇ

COMPARISON AND EVALUATION OF OBJECT ORIENTED SOFTWARE LANGUAGES WITH ANALYTIC HIERARCHY AND ANALYTIC NETWORK PROCESS

(M.Sc. Thesis)

Zelal ANIK

GAZI UNIVERSITY

INSTITUTE OF SCIENCE AND TECHNOLOGY

June 2007

ABSTRACT

The change in software technology has begun with the realization of the benefits of structural programming in 1970s and software development process has gained acceleration with the widening of object oriented programming in 1990s. Object oriented programming is more convenient to produce programs which are more understandable, better organized and easy to analyze, modify, debug. It is in brisk demand today due to many facilities it provides. The reason of preferring this study is the fact that object oriented programming will be the key point in programming methods in the coming years.

In this study, it is primarily aimed at determining the priorities among the features that a good programming language should have, finding a tool that will facilitate the choice of programming language and making this program well-arranged. But, choosing the proper programming language is a kind of decision making problem with multiple criterions which requires considering many factors. In this study, problem of the choice of object oriented programming language has been tried to be solved by Analytic Hierarchy Process (AHP) and Analytic Network Process (ANP), obtained results have been evaluated, compared and their compatibility with this process have been examined. No

study with ANP has been encountered for object oriented software languages in literature and with this aspect, this study contains a feature to be the first.

Science Code : 906.1.071
Key Words : Object oriented programming languages, Analytic Hierarchy Process, Analytic Network Process
Page Number : 103
Adviser : Assoc. Prof. Ö. Faruk BAYKOÇ

TEŞEKKÜR

Çalışmam boyunca bana rehberlik eden, her konuda yardımını esirgemeyen danışmanım Doç. Dr. Ö. Faruk Baykoç'a, değerli vakitlerini bana ayırarak çalışmama katkıda bulunma nezaketini gösterdikleri için Doç. Dr. M. Ali Akcayol'a, Dr. Ebru Sezer'e, Alev Mutlu'ya, Can Kaynak ve arkadaşına, zor zamanlarımda yanımda olan arkadaşlarıma, özellikle de hayatımın her anında sevgisini ve desteğini yanımda hissettiğim aileme tüm kalbimle teşekkür eder, minnetimi ve şükranlarımı sunarım.

İÇİNDEKİLER

Sayfa

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış bazı kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Kısaltmalar	Açıklamalar
AHP	Analitik Hiyerarşi Prosesi
ANP	Analitik Network Prosesi
ANSI	American National Standards Institute
BT	Bilişim Teknolojisi
CSC	Critical Success Chains
ÇKKV	Çok Kriterli Karar Verme
ÇÖKV	Çok Ölçütlü Karar Verme
ELECTRE	ELimination Et Choix Traduisant la Realite
ERP	Enterprise Resource Planning
IBM	International Business Machines Corporation
ISO	International Standards Organization
JVM	Java Virtual Machine
Rİ	Rassallık İndeksi
TO	Tutarlılık Oranı
TOPSIS	Technique for Ordered Preference by Similarities to Ideal Solution
VCL	Visual Component Library

1. GİRİŞ

Bilgisayar teknolojisindeki ilerlemelerin, son yıllarda baş döndürücü bir hıza erişmesi, beraberinde yeni çalışma alanlarını da gündeme getirmiştir. İnsanoğlunun bilgisayar teknolojisini ortaya koyuncaya kadar kullandığı araçların büyük bir çoğunluğu, kol gücüne dayanan çalışmaları kolaylaştırmaya yöneliktir. Özellikle II. Dünya savasından sonra yaşanan hızlı değişimler ve bu değişimlerden birisi olan bilgi teknolojisi, insanın günlük yaşamını, ekonomiyi, uluslararası ilişkileri, eğitimi etkilemiştir. Dünya, bu dönemde endüstri toplumundan bilgi toplumuna doğru hızlı bir değişim göstermiştir.

Bilgiye dayalı yeni ekonomide, ticaret başta olmak üzere birçok hizmetin elektronik ortama taşınıyor olması, yazılımın gelecekte de yıldızı parlak bir endüstri olacağının en belirgin göstergesidir. İlerideki yıllarda alış verişlerin büyük bir çoğunluğu elektronik pazar yerlerinden yapılacaktır. Kurumlar arasındaki ticari ilişkiler ağ üzerinden yürütülecektir. Bilgi ve belge akışı ağ üzerinden gerçekleşecektir. Tüm bunlar ancak yazılımla hayata geçebilecek gelişmelerdir.

Eğitimden üretime, ticaretten sanata kadar hemen her alanda BT(Bilişim Teknolojisi) yaşamımızın bir parçası olacaktır. Bu çerçevede hayata geçirilecek olan e-Devlet kavramı da, devletin birçok hizmetinin ağ üzerine taşınması anlamına gelmektedir. Bunun sonucu olarak BT kullanımı tüm bireylere kadar yayılacaktır. Evde, iş yerinde ve yolda, hemen her yerde ve her an BT'den yararlanılacaktır [1].

İş dünyasında internet ile birlikte uygulamalardan beklenenler de değişmiştir. Bu durum uygulama geliştirenleri doğrudan etkilemiştir. İnternet ortamında çalışabilen çok yönlü bir uygulama geliştirmek eski yöntemlerle aşırı derecede zaman ve insan gücü gerektirmektedir. Bu zorlukları aşmak için gelişen teknolojiye ve isteklere paralel olarak programlama dilleri ve program geliştirme araçları da gelişim içine girmiştir. Geliştirilen uygulamaların kapsamı ve boyutu genişlerken daha güçlü araçlarla daha büyük uygulamalar ortaya çıkmıştır.

Yazılım seçiminde birçok kriterin etkili olması problemin çözümü için Çok Kriterli Karar Verme yöntemlerinin kullanılmasını gerektirmektedir. Bu çalışmada bir programlama dilini seçerken dikkat edilen nitel kriterler ışığında nesne yönelimli dört yazılım dili arasında bir seçim yapılmaya çalışılmıştır. Programcılara ve akademisyenlere göre bir dillin yapısında barındırması gereken birçok nicel özellik önem derecelerine göre sıralanmıştır.

Çalışmanın birinci bölümünde karar verme süreci incelenmiş ve çok kriterli karar verme yöntemlerinden bahsedilmiştir. İkinci bölümde çok kriterli karar verme yöntemlerinden biri olan Analitik Hiyerarşi Prosesi (AHP) ve kriterler arası ilişkileri de kontrol edebilen Analitik Network Prosesi(ANP) ile ilgili detaylı bilgi verilmiştir. Üçüncü bölümde ise programlama dillerinin gelişiminden, yazılım geliştirme ve bu alandaki yaklaşımlardan, yazılım sektörünün yapısal farklılıklarından bahsedilmiştir. Dördüncü bölümde ise problem hem AHP hem de ANP yapısına göre modellenip Superdecisions programı ile çözülmüş, bu çözümler incelenip, karşılaştırılmıştır. Son bölümde ise genel bir değerlendirme yapılmış ve önerilerde bulunulmuştur.

2. KARAR VERME SÜRECİ VE ÇOK KRİTERLİ KARAR VERME

Karar verme, hedef ve amaçların gerçekleştirilmesi yönünde alternatif eylem planlarından birini seçme sürecidir. Şirketlerin işlerinin devamlılığını sağlayabilmeleri, verimliliğini artırabilmeleri, gelecek yıllarda başarıyı yakalayabilmeleri ya da işlerini çeşitlendirerek büyütebilmeleri, doğru kararları alabilmelerine bağlıdır. Karar verme tüm yönetim fonksiyonlarının özünü oluşturur. Karar verme bir işletmenin temel taşlarından biridir. Doğru kararların alınması rekabetçi avantaj kazanmak ve sürdürmek için daima gereklidir.

2.1. Karar Analizi Gereksinimi

Karar verme herkesin profesyonel ve kişisel hayatının önemli bir parçasıdır. Hepimiz birer karar vericiyiz ve karar vermek için uygulanan tekniklerden faydalanabiliriz. Karar analizi birkaç bilim dalı ile ilgili bir disiplindir. Karar analizi aynı zamanda spesifik bir kararı inceleme sürecidir.

Karar verme, elde edilen ile vazgeçilen değerler arasında denge kurulması durumu nedeniyle zordur. Bazen bir amacı en iyi şekilde karşılayan alternatifi seçmek başka bir amaçtan ödün vermeyi gerektirir. Bazı kararlar ise çok fazla faktörün göz önüne alması nedeniyle zordur. Karar vermeyi zorlaştıran diğer iki faktör ise endişe ve oybirliğine varamamaktır [2].

Geçmişte yöneticiler, kararlarını bugüne göre daha sınırlı bilgi ile genellikle deneyim, yargı ve sezgilerine göre vermekteydiler. Bugün tecrübe, sezgi ve yargı gibi subjektif unsurlar rasyonel karar almak için gerekli ancak yeterli değildirler. Karar verme süreci günümüzde çeşitli faktörlerin etkisiyle giderek karmaşıklaşmaktadır. Bu durum karar verme sürecine analitik gözle bakmayı gerektirmektedir.

Karar verme bir takım problemleri çözerken yeni problemler yaratan dinamik bir süreçtir. Belli bir konuda organizayonun amaçlarını destekleyen spesifik

düzenlemeler başka alanlarla çatıştışabilir. Bu süreç genel olarak aşama aşama ilerleyerek optimize edilir [3].

Karar analizi, karar problemlerinin matematiksel modelini ortaya koyup, sayısal ve istatistiksel irdelemelere bağlı olarak hareket tarzı öneren bir yöntem olarak tanımlanabilir Tüm diğer modeller gibi karar analizi modelleri de gerçeği ancak yaklaşık olarak yansıtırlar. Karar analizi modellerinden beklenmesi gereken, soruna ilişkin doğru ve kesin cevabı vermesi değil, aslında çok daha önemli olan problemin önemli özelliklerine derinlemesine yaklaşım getirmesidir. Yapılan çeşitli duyarlılık analizleri ile herhangi bir varsayımdaki küçük bir değişmenin alınan kararı nasıl değiştireceği veya bu değişimlere kararın duyarsız kalıp kalmayacağı irdelenebilir [4].

2.2. Karar Verme Süreci

Karar verme süreci, karar vermek için kullanılan teknik ve yöntemlerin eylem düzenini ve izlenen yolu ifade etmektedir. Karar verme süreci ile ilgilenen bir bilim dalı olan Karar Teorisi, ekonomi, istatistik, felsefe, psikoloji, idari bilimler ve yöneylem araştırması gibi farklı disiplinlerden önemli ölçüde etkilenmiştir. Karar teorisinin öncelikli hedefi, karar verme sürecindeki belirsizliği ve karmaşıklığı azaltmak suretiyle olası alternatiflerin sistematik biçimde değerlendirilmesi için karar vericilere yardım etmektir.

Karar verme süreci bir çok araştırmacı tarafından kendi ilgi alanlarına göre incelenmiştir. Glibert ve Israel, karar süreci aşamalarını aşağıdaki şekilde detaylı olarak ifade etmişlerdir [5]:

1. Amacın belirlenmesi,
2. Kontrol edilebilen değişkenlerin belirlenmesi,
3. Kontrol edilemeyen değişkenlerin belirlenmesi,
4. Kısmi kontrol edilebilen değişkenlerin ve kısmi kontrol edilebilen değişkenlerle kontrol edilebilen değişkenler arası ilişkilerin belirlenmesi,

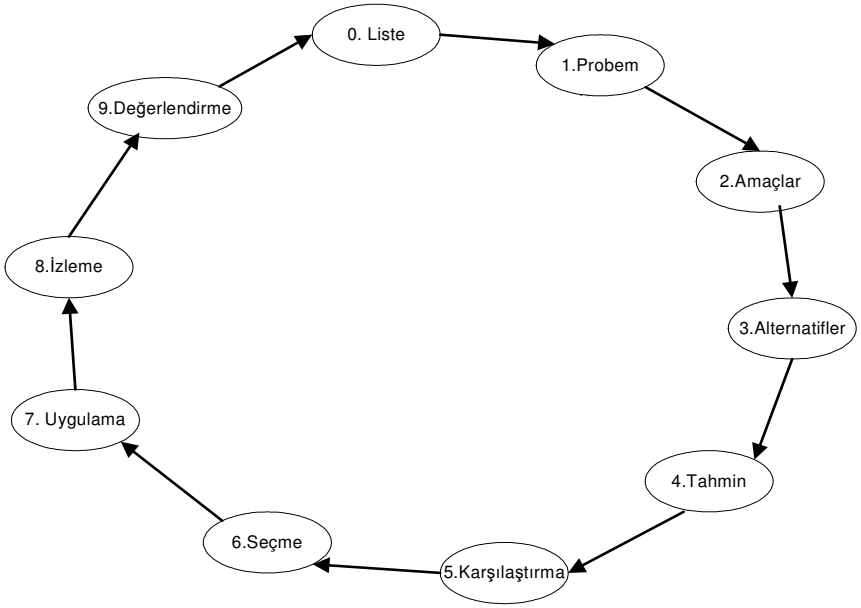
5. Amaca bağlı olarak her bir olası kararın etkisinin belirlenmesi,
6. Kararın verilmesi,
7. Sonuçların yorumlanması,
8. Sonraki çalışmalar için karar sürecinin yinelenmesi.

Hicks , karar verme sürecinin aşamalarını Glibert ve Israel'den farklı olarak beş aşamada ifade etmiştir [5]:

1. Kararın amaçlarının belirlenmesi,
2. Amaçları karşılayan alternatif yolların bulunması,
3. Değerlendirme kriterlerinin/tekniklerinin belirlenmesi,
4. En iyi eylem tarzının seçimi,
5. Seçilen eylem tarzının uygulanması.

2.3. Karar Vermeye Rasyonel Yaklaşım

Rasyonel model, karar vermeye kapsamlı ve sistematik bir yaklaşım sağlar. Her adım temel bir faaliyeti gösterir ve bir önceki adımın üstüne kurulur. Rasyonel modelin 10 adımı Şekil 2.1'de görülmektedir [2].



Şekil 2.1. Karar vermeye rasyonel yaklaşım [2]

Gündem Listesi Oluşturma

Herhangi bir karar analizini uygulamadan önce hangi problemlerin üzerinde çalışılacağına karar vermek gerekir. Hangi problemlerin hangi sırada ele alınacağı, göreli önemleri belirlenen zaman dilimi içinde tartışılır.

Problemi Tanımlama

Problemin yapısı ve kapsamı belirlenir. Problemin geçmişi, nedenleri, bu problemi ve benzeri problemleri çözmek için geçmişte yapılmış çalışmaların sonuçları belirlenir. Problemin yapısı akış diyagramı ile gösterilecekse burada belirtilir.

Amaçlar

Problem birden çok ortağı etkiliyorsa bu ortaklar ve temel öncelikleri tanımlanır. Tüm amaçlar tanımlanır. Tek bir amaç içeren problemler için bu bölüm problemi tanımlama bölümüne dahil edilebilir.

Alternatifler

Her bir alternatif tanımlanır. Alternatif listesinin nasıl çeşitlendirilebileceği, kapsamlı hale getirilebileceği tartışılır. "En iyi" çözümün analize dahil edilmesi önemlidir çünkü seçim göz önüne alınan alternatifler arasından yapılacaktır.

Tahmin

Tahminlerin nasıl oluşturulduğu tanımlanır. Her bir alternatif için uygulandığı takdirde neler olabileceği tanımlanır. Kesin olmayan olaylar ve bunların ortaya çıkma olasılığı tartışılır. Tahminlerin güvenilirliği tartışılır. Tahmin edilen her parametre için güven aralığı belirlenir.

Karşılaştırma

Tek amaçlı problemler için: İlgili karşılaştırmalar yapılarak en iyi puanı alan alternatif belirlenir.

Çok amaçlı problemler için: Her bir alternatifin her bir amacı ne kadar gerçeklediğini gösteren bir tablo hazırlanır. Elde edilen ile vazgeçilen değerler ararsında denge kurulması, uygunluk durumu açıklanır.

Seçme

Gerçekleştirilen duyarlılık analizleri açıklanır. En iyi alternatif, tavsiye edilen alternatif ya da alternatifler ve niçin tavsiye edildikleri açıklanır.

Uygulama

Bir alternatif seçildikten sonraki başarı uygulama planının ne kadar iyi yürütüldüğüne bakar. Uygulama süreci 4 adımda gerçekleştirilir :
Girdi→ Süreç → Çıktı → Etki

İzleme

İzleme çalışma durumunun takip edilmesidir. Bu uygulama sürecinin değiştiği her adımını kontrol etmeyi gerektirir. "Yeterli kaynak tahsis edildi mi?", "Uygun prosedür takip edildi mi?", " Yeterli çıktı yaratıldı mı?", "Beklenen etki oluşturuldu mu?" gibi sorulara cevap aranır.

Değerlendirme

Değerlendirme orijinal problemin çözüldüğü kapsamın incelendiği, hareket yönünün değerlendirildiği ve şayet gerek görülüyorsa ne kadar düzeltici ek faaliyetin uygun olacağının görüşüldüğü aşamadır.

2.4. Karar Modelleri

Karar verme konusunda kullanılan modeller arasındaki farklılık ortaya çıkması beklenen olaylara göre yapılır ve karar verenin olaylar hakkındaki bilgi derecesini yansıtır. Olaylar ve gerçekleşme olasılığı arasındaki ilişkiyi tanımlayan bu farklılık aşağıdaki gibidir [5]:

1. Belirlilik halinde karar verme,
2. Risk halinde karar verme,
3. Belirsizlik halinde karar verme,
4. Kısmi bilgi halinde karar verme,
5. Rekabet halinde karar verme .

2.4.1. Belirlilik halinde karar verme

Tüm gerçeklerin bilindiği varsayımı halinde belirlilik halinde karar verme söz konusudur. Gerçekte, böylesi bir belirlilik de oldukça ender görülür. Ancak, böylesi bir tam bilgi varsayımından hareketle, bir araştırmaya verebilecek bedelin üst sınırının saptanması bazen yararlı olmaktadır.

Belirlilik halinde, her bir seçime ilişkin olarak tam bilgi vardır. Karar verici, gelecek ve sonucu konusunda güvenceli bilgiye sahiptir. Belirlilik, karar vericinin haberdar olma durumunu yansıtır. Karar verici, amacına en uygun olan alternatifi kolayca seçebilir. Dolayısıyla en büyük kazanç değeri amacın en iyi başarılma derecesi olur ve karar kriteri en büyük kazancın seçimidir [5].

2.4.2. Risk halinde karar verme

Her bir seçeneğin belirli bir sonuca götüreceğinin bilindiği, ancak karar verici tarafından bilinen bu sonuçların birer olasılık olduğu ortamdır Risk kavramında önemli olan ölçülebilme ve kestirilebilmedir. Bir işletmede çeşitli nedenlerle ortaya

çıkacak risklerin doğuracağı zararların önceden kestirilebilmesi için risk planlaması yapılmalıdır ve riskin maliyeti hesaplanmalıdır.

2.4.3. Belirsizlik halinde karar verme

Belirlilik halinde karar vermede, bir kararı izleyen çıktıların gerçekleşme olasılıkları belirli değildir. Böylesi tümü ile belirsiz bir ortama, gerçek hayatta ender rastlanılır. Genellikle konu ile ilgili yöneticilerin ön yargılarına dayanılarak bu olasılıklar kestirilebilir. Bu durumda ise risk altında karar verme sorunu ortaya çıkar. Ayrıca, geçmiş devrelerin kayıtlarına dayanılarak da, çıktıların olasılıkları tanımlanabilir. Böylesi bir yaklaşım yanlı kişisel yargılara bağlı olarak ortaya konan olasılıklardan daha tutarlı sonuçlar verebilir [5].

2.4.4. Kısmi bilgi halinde karar verme

Olasılık dağılımının şekli (Normal, Poisson, Binominal, vb.) bilindiği zaman ve dağılımın parametreleri ile karakteristikleri (Ortalama, Mod, Medyan veya simetrik ölçüleri; Çarpıklık ve Basıklık) hakkında bilgi varsa, karar problemi yalnız kısmi bilgiler ile karar vermeyi gerektirir. Risk halinde karar problemlerinde karar verici, en iyi tahminin bulunduğu ön olasılıklara sahiptir. En iyi karar için olaylar hakkında ek bilgiler istenebilir. Bu yeni bilgiler düzeltilebilir veya olaylar hakkında daha geçerli olasılık tahminlerine dayalı son kararın verilebilmesi için ön olasılıklar güncelleştirilir [5].

2.4.5. Rekabet halinde karar verme

İşletmelerin rakipleri vardır. Dolayısıyla, bir işletmenin iç problemlerine en iyi çözümleri bulması, işletme gelişimi açısından yetmeyecektir. Üstelik bu defa, çoğu zaman kontrolü dışında olan rakiplere göre kendisini ayarlaması ve rakipleri karşısında kendisine mümkün olan en çok getiriyi sağlayacak stratejiyi saptaması gerekecektir [5].

2.5. Çok Kriterli Karar Verme (ÇKKV) Yöntemleri

ÇKKV, bir karar vericinin sayılabilir sonlu ya da sayılamaz sayıda seçenekten oluşan bir küme içinde en az iki kriter kullanarak yaptığı seçim işlemi ya da diğer bir deyişle, iki veya daha çok kritere dayalı değerlendirme yaparak alternatifler arasından seçim yapması olarak tanımlanabilir.

ÇKKV'de nihaî karar, kriterler arası ve kriterler içi karşılaştırmalara dayanır. Kriterler arası karşılaştırmada, kriterler birbirleriyle kıyaslanırlar. Bu kıyaslamadan amaç, kriterleri bir öncelik sırasına sokmak, başka bir ifadeyle, kriterlerin karar verici için önem derecelerini belirlemektir. Kriterler içi kıyaslama ise, belirli bir kriter esas alındığında, hangi alternatifin o kriterde daha cazip olduğunu tespit etmek için yapılır. Son karar, bu iki kıyaslamanın sentezi sonucunda verilir[6].

2.5.1. ÇKKV yöntemlerinin sınıflandırılması

ÇKKV yöntemleri çeşitli şekillerde sınıflandırmaktadırlar. Seçenek sayısına göre şu şekildedir [7]:

Çok Amaçlı Karar Verme(Multi-Objective Decision Making): Seçeneklerin bir matematiksel programlama yapısı ile dolaylı olarak tanımlandığı ve sonsuz sayıda olduğu sürekli durumlarda kullanılır.Bir tasarım problemidir ve matematiksel optimizasyon teknikleri gerektirir. Hedef programlama, tamsayılı çok amaçlı programlama, dinamik programlama bu bölüme dahil edilebilecek yöntemlerdir.

Çok Ölçütlü Karar Verme (Multi-Attribute Decision Making): Seçeneklerin sonlu sayıda olduğu ve listelenebildiği kesikli durumlarda kullanılır. Bir tasarım probleminden çok seçim problemidir. Matematiksel optimizasyon araçları gerektirmeyebilir. Puanlama modelleri. AHP, ANP, TOPSIS (Technique for Ordered Preference by Similarities to ideal Solution), ELECTRE (ELimination Et Choix Traduisant la REalite) bu grupta sayılabilecek yöntemlerdir.

2.5.2. ÇKKV yöntemlerinin karakteristik özellikleri

Bütün ÇKKV problemleri aşağıda belirtilen ortak özellikleri paylaşmaktadırlar [7].

Alternatifler: Yüzlercesi arasından sınırlı sayıdaki alternatifler ayıklanır, önceliklendirilir, seçilir ve/veya sıralanır. Örneğin binlerce başvuru arasından birkaçının seçilmesi gibi.

Çok kriterlilik: Her problem birden fazla kritere sahiptir. Her problem setinde ilgili kriterler belirlenir. Kriterlerin miktarı problemin doğasına bağlıdır. Karar için düşünülmesi gereken yüzlerce faktör olmasına rağmen karar verici, en önemlilerini kriter olarak kabul edebilir.

Aynı birimle ölçülme: Her kriter farklı ölçüm birimlerine sahip olabilir. Bir otomobil seçiminde yakıt tüketimi litre/km olarak ifade ederken, satış fiyatının dolar olarak ifade edilmesi gibi. Güvenlik ise sayısal olmayan yollardan ifade edilir. Sağlıklı bir karar alabilmek için bütün bu ölçüm farklılıklarının giderilmesi gerekir.

Kriter ağırlıkları: Hemen hemen bütün ÇKKV yöntemleri, her kriterin göreli önemini bulabilmek için bilgiye ihtiyaç duyar. Ağırlıklar direkt karar verici tarafından belirlenebileceği gibi daha sonra açıklanacak olan yöntemlerle de bulunabilir.

Karar matrisi: ÇKKV problemleri basit olarak bir matris formatında ifade edilebilir. Burada sütunlar, verilen problemdeki kriterleri, satırlar ise alternatifleri belirtir.

İşletmelerde genel olarak analiz sonuçlarını, sezgisel olarak değerlendirilmektedir. Araştırmalar, pek çok günlük kararın sezgisel olarak alınmasının yeterli olmasına rağmen, karmaşık ve hayati kararlar için bu yolun tek başına yeterli olmadığını göstermektedir. Modern karar destek yöntemlerini kullanan işletmeler, globalleşen iş ilişkilerine öncülük etmekte ve bu ilişkiler ağını yönetmekte rekabetçi avantaj sahibi olabilmektedirler. Son yıllarda kullanımı gittikçe artan modern karar destek

yöntemlerinden ikisi, karar verme sürecinde kriterler arasındaki ilişkileri dikkate alan Analitik Network Prosesi(ANP) ve kriterler arasında tek yönlü ilişkiyi esas alan Analitik Hiyerarşi Prosesi(AHP)'dir.

3. ANALİTİK HİYERARŞİ PROSESİ VE ANALİTİK NETWORK PROSESİ

AHP kavramsal karşılaştırmalar ile gerçek hayatta çok karşılaşıldığını ve bu nedenle bir çeşit matematiksel yaklaşımın kullanılmasının söz konusu kararlaştırma işlemini kolaylaştıracağı düşüncesine dayanmaktadır. AHP nitel ve nicel faktörleri birleştirme olanağı sunan güçlü ve kolay bir yöntemdir.

ANP, AHP'nin çok genel bir formudur. AHP'nin esasları, kıyaslama mantığı ANP için de kullanılmaktadır ancak, ANP ile karar verme sürecinde faktörler arasındaki ilişkiler de dikkate alınmaktadır.

3.1. Analitik Hiyerarşi Prosesi (AHP)

Bu bölümde AHP ile ilgili literatür araştırmasının ardından, AHP'nin teorik temelleri hakkında genel bilgiler verilmiş, AHP'yi oluşturan önemli tanımlar, aksiyomlar, ölçme, tutarlılık oranları ile yöntemin matematiksel açıdan incelenmesi yapılmıştır.

AHP, 1970'lerin sonlarında Saaty tarafından geliştirilmiştir. AHP'nin karar vericiler tarafından tercih edilmesinin nedeni çok kriterli kararların alınmasında sübjektif kriterleri dikkate alabilmesidir. ÇKKV yaklaşımlarından olan AHP'de nitel faktörler başlıca öneme sahiptir. Alternatiflerin ayrıntılı değerlendirmesinde nitel ve sayısal faktörleri birleştirebilen bir tekniktir. AHP çeşitli seviyelerde birbirinden bağımsız olan faktörlerin, içinde bulundukları hiyerarşik yapıda değerlendirilmesinde kullanılır.

AHP, insanoğlunun karmaşık bir problemi nasıl algılayıp biçimlendirdiğini gözler önüne seren bir modeldir ve çeşitli gözlemler sonucunda oluşturulmuştur. Bu gözlemlerden biri, kişilerin bu tip durumlarda öğeleri gruplandırıp problemi hiyerarşik olarak parçalara ayırma işlemidir. Söz konusu parçalama kişiden kişiye farklılık gösterebilir; ancak kişiler bir sorunu aynı şekilde yargılıyorlar ise çözüm de yaklaşık aynı olacaktır [4].

3.1.1. Literatür araştırması

Literatürde AHP tekniği kullanılarak yapılmış çok fazla çalışma vardır. Çalışmanın bu bölümünde özellikle bilişim teknolojileri ile ilgili olan AHP çalışmalarından bahsedilecektir.

Ossadnik W., Lange O. çalışmalarında AHP'yi destekleyen üç yazılım ürününün kalitesini yine AHP tekniğinden faydalanarak değerlendirmişlerdir. Çalışmada kurulan modelle ilgili kriterlerde uluslararası ISO/IEC 9126 normlarından faydalanılmışlardır [8].

Zhu, X., Dale, A. P. çalışmalarında bir çeşit AHP Programı olan JavaAHP programının temel özelliklerini anlatmış ve optimum inşaat alanını seçimiyle ilgili örnek bir uygulama yapmışlardır [9].

Gönenç D. S. yüksek lisans çalışmasında yazılım geliştiricilere ve proje liderlerine kendi yazılım takımlarının performansını iyileştirebilmeleri bazı öneriler sunmuştur. Bu amaçla çalışmada performansı ve motivasyonu etkileyen faktörleri belirlemek için "Delphi Tekniği"ni kullanmış, AHP tekniğinden faydalanarak kaynakların dağıtım katsayılarını hesaplamış, "Expert Choice" yazılımı kullanılarak ilgili değerlendirmeleri yapmıştır [10].

Vincent S. Lai ve arkadaşları çalışmalarında en uygun Çoklu Ortam Yetkilendirme Sistemi(Multi-Media Authorizing System, MAS)'nin seçimi için AHP tekniğini kullanarak bir örnek olay incelemesi yapmışlardır. Üç tane Çoklu Ortam Yetkilendirme Sistemi (MAS) ürünü belirlenmiş ve AHP tekniğini kullanarak ürünler arasında sıralama yapmıştır [11].

Yücel D. Ve Erkut H. yaptıkları araştırmada AHP'ye uygun bir modelle bilişim teknolojilerinin yönetim kalitesi parametreleri üzerinden çalışma yaşam kalitesini nasıl etkilediği incelemişlerdir [12].

Younghwa Lee , Kenneth A. Kozar ve Smith, J. M çalışmalarında AHP yaklaşımı ile en ideal web sitesini seçebilmek için web sitelerini etkileyen kalite faktörlerini ve göreli önemlerini incelemişlerdir [13].

Chun-Chin Wei, Chen-Fu Chien, Mao-Jiun J. Wang çalışmalarında uygun ERP programının seçilmesi için AHP yaklaşımı ile kapsamlı bir çerçeve sunmuşlardır[14].

Karar vericiler AHP'de değerlendirme skorlarını belirlemede çoğu zaman emin değildirler. Bulanık AHP bu zorluğu ortadan kaldırabilmektedir. Başlıgil H. çalışmasında üç yazılım tipini incelemiş ve müşterilerin yazılım seçim sürecinde dikkate aldığı önemli kriterleri belirlemiştir. Bu kriterlere göre yazılımları karşılaştırmak ve en iyisini seçmek için Bulanık AHP kullanmıştır [15].

Ngai, E.W.T. ve Chan E.W.C. çalışmalarında bilgi yönetimini destekleyecek en uygun aracı seçmek için bir AHP uygulaması yapmışlardır. AHP modeli kurulmuş ve Hong Kong'da öncü bir firmada uygun bilgi yönetimi aracını seçebilmeleri için karar vericileri destekleyecek şekilde gerçek duruma uygulanmıştır [16].

BT projeleri başarısız olma ihtimalinin yüksek olması nedeniyle diğer mühendislik projelerinden farklı ve potansiyel olarak daha zordur. Bu nedenle projenin başarılı olma ihtimalini yükselten kritik başarı faktörlerini belirlemek önemlidir. Repiso L. R., Setchi R. ve Salmeron J. L. çalışmalarında IT projelerinin kritik başarı faktörlerini belirlemek, gruplandırmak ve değerlendirmek için üç yöntem önermişlerdir: Kritik Başarı Zincirleri (Critical Success Chains -CSC), AHP, Bulanık Bilişsel Haritalar (Fuzzy Cognitive Maps -FCM). Bu metotları IT projelerinin başarısı ile ilgili olarak avantaj, dezavantaj ve kısıtlarına göre karşılaştırarak en uygun yöntemi belirlemeye çalışmışlardır [17].

3.1.2. AHP'nin aşamaları

AHP'nin aşamaları aşağıda açıklanmıştır [18] :

Problemin Tanımlanması: Bu aşama, aslında diğer bütün karar verme yöntemlerine ait problemlerinin çözümünde de kullanılan ilk aşamadır. Problemin tanımlanması sırasında dikkat edilmesi gereken en önemli husus, bu problemin AHP yöntemine uygun olup olmadığı, diğer bir deyişle, elemanların kantitatif göstergeleri bulunup bulunmadığıdır. AHP yönteminin en önemli özelliği, öznel değerlendirmeler için bir ölçü birimi yaratmasıdır.

Sistemin Gözlenmesi: AHP çok amaçlı, karmaşık bir problemi, her düzeyi belirli kriterlerden oluşan bir hiyerarşiye ayrıştırır. Bu kriterler daha sonra alt elemanlara bölünürler. En alt düzeye ise, değerlendirilecek olan seçenekler yerleştirilir. Böyle bir hiyerarşik yapının kurulabilmesi ve söz konusu kriterlerin belirlenebilmesi için sistemin bütünü, elemanları ve bunların birbirleri ile ilişkileri iyice gözlenmelidir.

Hiyerarşik Yapının Kurulması: Bu aşama, klasik problem çözme teknikleri ile karşılaştırıldığında daha çok "model kurma" aşamasına karşılık gelmektedir. Ancak bu model kişiden kişiye değişiklik gösterir ve bunlardan birinin doğru, diğerlerinin yanlış olması söz konusu değildir. Mantıklı bir subjektif yaklaşım, çoğu zaman objektif yaklaşımlardan daha sağlıklı olmaktadır. Hiyerarşide en önemli husus, her bir seviye elemanları ve bu elemanlar arasındaki ilişkilerdir. Çünkü bu model sayesinde, her seviyedeki elemanların göreli gücünü (hiyerarşik modelin en üst seviyesine yaptığı etkiyi) ölçmek asıl amaçtır.

Önceliklerin Belirlenmesi (İkili Karşılaştırmalar): Model kurulduktan sonraki aşama, aynı hiyerarşi düzeyindeki faktörlerin göreli ağırlıklarının belirlenmesidir. Bu işlem, bir üst düzeydeki faktörle bağlantılı olan alt düzeydeki faktörlerin, kendi aralarında yapılacak ikili karşılaştırmaları şeklinde gerçekleştirilir.

Sentez: Hiyerarşinin en alt düzeyinde değerlendirilecek seçenekler bulunmaktadır. Dolayısıyla önceliklerin belirlenmesi işleminin benzeri olarak, seçeneklerin her alt kriter bazında ikili karşılaştırmaları yapılmıştır. Bütün ağırlıkların birleştirilmesi sonucu seçeneklerin genel ağırlıkları bulunur.

Değerlendirme ve Sonuç: Özvektörün hesaplanması sırasında problemin rassal indeksi hesaplanır. Bu indeksin 0.1 ve daha yüksek çıktığı durumlarda değerlendirmelerin tutarsız olduğu belirtilmektedir. Dolayısıyla, elde edilen sonuçlar sağlıklı seçim yapılabilmesi için yeterli olmadığından sitemin daha kararlı hale getirilmesinde veya yeni hedeflere yönelmede geri besleme olarak kullanılabilir. Hiyerarşinin yapısını değiştirmek suretiyle yapılabilecek model değişiklikleri aşamasına geçmeden önce, ikili karşılaştırmalar kontrol edilmelidir. Belki de önceliklerde yapılabilecek bazı düzeltmeler, problemin genel uyumsuzluk indeksini düşürmeye yetecektir. Söz konusu indeks kabul edilebilir oranda ise, mantıklı olarak en büyük göreli ağırlığa sahip olan alternatif seçilir ve uygulanır.

3.1.3. AHP' nin uygulama alanları

AHP çeşitli endüstri kuruluşlarında, resmi kuruluşlarda ve birçok alanda başarıyla uygulanmıştır. Başlıca uygulama alanları aşağıda sunulmuştur[19, 20].

<u>Ekonomi ve yönetim problemleri</u>

- Performans analizi
- Veri tabanı seçimi
- Ürün tasarımı
- Muhasebe/Finans
- Sermaye yatırımı
- Karar destek
- Üretim
- Politika/Strateji
- Pazarlama

- Risk analizi
- Strateji planlama
- Kaynak tahsisi
- Makro ekonomik tahminler
- Kaynak seçimi
- Tesis yeri seçimi
- Grup karar verme
- Ulaştırma
- Tahminler

Politik problemler

- Silah kontrolü
- Politik adaylık
- Global etkiler
- Güvenlik sistemlerinin değerlendirilmesi
- Uluslar arası görüşmeler ve çelişkiler
- İş ve halk politikaları
- Komplo teorileri

Sosyal problemler

- Rekabet ortamı
- Eğitim
- Hukuk
- Çevresel etkiler
- Tıp
- Sağlık
- Nüfus dinamikleri
- Kamu sektörü

Teknolojik problemler

- Pazar seçimi
- Portföy seçimi
- Teknoloji transferi
- Bilgisayar ve bilgi seçimi
- Uzay araştırmaları

3.1.4. AHP' nin avantajları

Saaty'ye göre AHP'nın esas özellikleri ve avantajları [19, 21] :

1- *Model tekliği ve benzersizliği (Unity):* AHP birçok karar verme problemine uygulanabilecek nitelikte anlaşması kolay olan esnek bir yöntemdir.

2- *Karmaşıklık (Complexity):* AHP karar verme sürecinde kullanılan faktörlere ilişkin hem yerel hem de global ağırlıkları incelemeye fırsat verir.

3- *Bağımlılık (Interdependence):* AHP'de tek yönlü bir bağımlılık söz konusudur.

4- *Hiyerarşik yapılanma (Hierarchy Structuring):* AHP karar verme problemlerinin hiyerarşik yapılanmasında birinci seviyede amaç, ikinci seviyede faktörler, üçüncü seviyede de alternatifler yer alır.

5- *Ölçme (Measurment):* AHP, karar verme sürecinde kullanılan faktörleri ikili karşılaştırmalar kullanarak ölçer ve her faktör ve alt faktör için bir ağırlık değeri hesaplar.

6- *Uyumluluk (Consistency):* AHP, karar verme sürecinde kullanılan ikili karşılaştırma karar matrisinin tutarlığını inceler ve daha hassas ve mantıklı sonuçlar alınmasını sağlar.

7- *Birleştirmek (Synthesis) :* AHP her alternatif için bir öncelik değeri hesaplar.

8- *Ödünleşim (Tradeoffs):* AHP, karar verme sürecinde kullanılan faktörlere bağlı olarak alternatif önceliklerini belirler ve sonucunda bu öncelikleri birleştirir.

9- *Yargı ve Grup uyumu (Judgment and Consensus):* AHP karar verme sürecinde birden fazla karar vericinin yargılarını birleştirmeye imkan sağlar.

10- *Sürecin Tekrarı (Process Repetition):* AHP karar vericilerinin yargılarını karar verme sürecinde değiştirmesine imkan sağlayan esnek bir yöntemdir, ayrıca karar verme sürecinde kullanılan faktör ve alt faktörlerin değiştirilmesine de imkan sağlar.

3.1.5. Hiyerarşik yapının oluşturulması

Bir hiyerarşi, kompleks bir problemin çoklu seviyeli bir yapıda gösterilmesidir. Söz konusu yapının ilk seviyesinde amaç bulunur ve onu kriterler ve alt kriterler takip eder. Hiyerarşik yapının en alt seviyesinde ise alternatifler bulunmaktadır. Hiyerarşik yapı elde bulunan tüm faktörlerin belirlenen hedef altında birbirleri ile olan ilişkilerini gösteren nicel bir yöntemdir. Bir hiyerarşide mutlaka belirli bir düzeydeki bir öğenin, o düzeyin bir altındaki tüm öğelerle ilişkili olması gerekmez Öte yandan hiyerarşi bir karar ağacı da değildir. Her düzey probleme ilişkin farklı bir kesiti yansıtabilir.

Bir hiyerarşik yapının oluşturulması esnasında dikkat edilmesi gereken önemli hususlar aşağıda verilmiştir [18].

- Hiyerarşik yapı problemi en iyi şekilde temsil etmelidir
- Problemi etkileyen tüm yan faktörler göz önüne alınmalıdır,
- Çözüme ışık tutabilecek tüm yayın ve belgeler dikkate alınmalıdır,
- Problemin içerisinde rol alacak katılımcılar belirlenmelidir.

Hiyerarşik yapının oluşturulmasında temel adım, büyük ölçekli bir sistemin alt sistemlere bölünmesidir. Hiyerarşi, genel ve az kontrol edilebilen faktörden, daha spesifik ve kontrol edilebilen faktöre doğru yapılmalıdır. Ayrıca bir hiyerarşi problemi

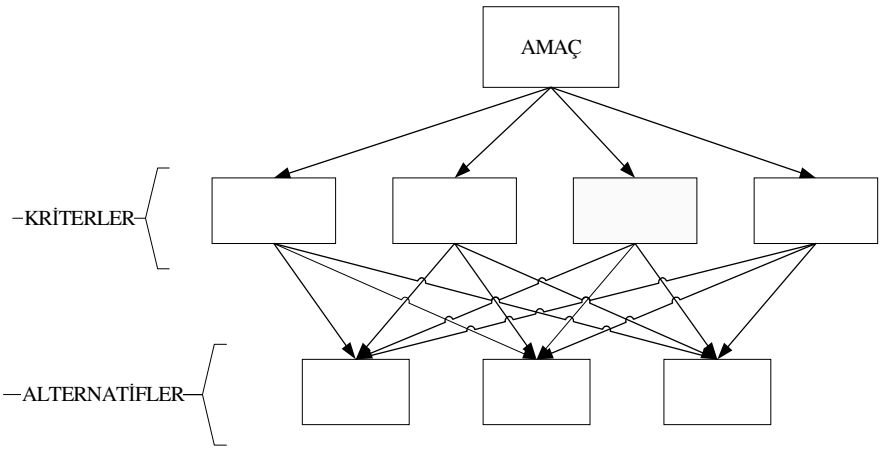
temsil edebilecek kadar büyük, öğeler üzerindeki değişikliklere tepki verebilecek kadar da küçük olmalıdır [18].

Hiyerarşik yapının oluşturulması, problemin daha küçük parçalara ayrılarak incelenmesi için sistematik bir prosedürün oluşturulabilmesine imkan verecektir. Hiyerarşik yapı, sistemi oluşturan tüm seviye veya bileşenlerin aralarındaki fonksiyonel bağımlılığın, sistem geneli üzerindeki etkisini en iyi ifade eden yapıdır. Ancak bu sayede çok karmaşık problemler basitleştirilebilmekte ve neden sonuç ilişkisini ortaya koyan doğrusal zincir yapısı oluşturulabilmektedir. Bu yaklaşımın diğer bir etkisi de, hiyerarşinin üst seviyesi ile alt seviyeleri arasındaki bağımsızlığı belirlemesidir. Bir hiyerarşik yapıdaki her seviye, problemin farklı aşamalarını gösterebilir[18].

Alt ve üst seviyedeki elemanların birbirleriyle etkileşimine göre tam ve tam olmayan şeklinde iki çeşit hiyerarşi modeli vardır.

Tam Hiyerarşi Modeli

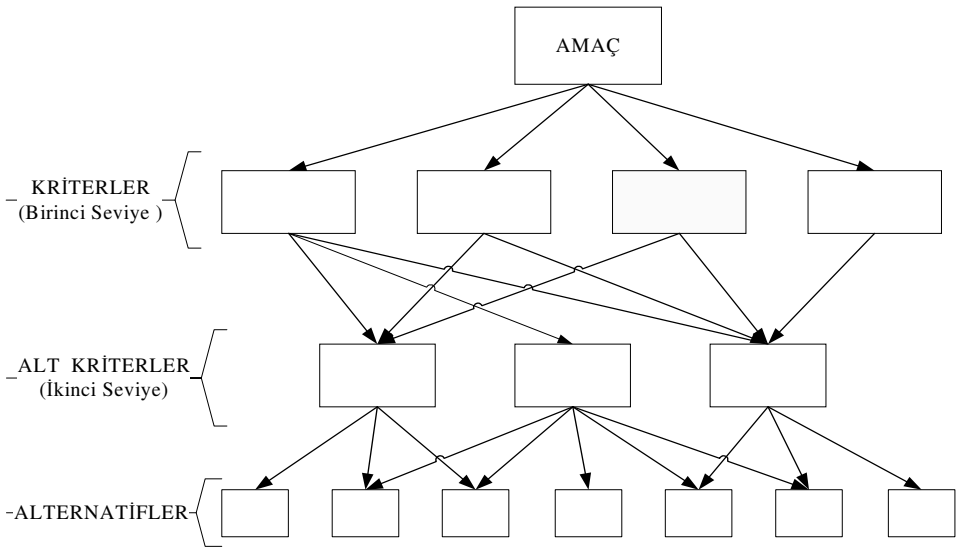
Şekil 3.1'deki gibi bir düzeydeki elemanların bir alt düzeydeki elemanların hepsiyle etkileşim halinde olduğu modellerdir.



Şekil 3.1. Tam hiyerarşi modeli

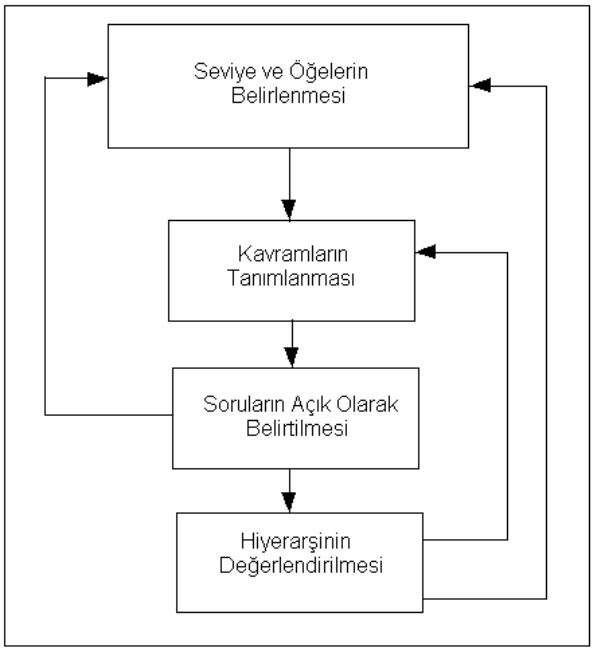
Tam Olmayan Hiyerarşi Modeli

Şekil 3.2'deki gibi bir düzeydeki elemanların bir alt düzeydeki elemanların hepsiyle etkileşim halinde olmadığı modellerdir.



Şekil 3.2.Tam olmayan hiyerarşi modeli [22]

Hiyerarşinin oluşturulması, birbiriyle ilgili olan ve ardışık olmayan üç prosesi gerektirmektedir. Bunlar: Seviye ve öğelerin belirlenmesi, kavramların tanımlanması ve son olarak da soruların açık olarak belirtilmesidir. Şekil 3.3.'te bu üç bileşen arasındaki ilişki gösterilmektedir.



Şekil 3.3. Hiyerarşinin oluşturulması [18]

Karar vericinin soruların cevaplanması ile ilgili bir sorunu olduğu takdirde, ilk iki bileşenin revize edilerek düzeltilmesi gerekecektir. Sorulardaki belirsizlikler karar vericiyi yanlış kriter veya alternatifi seçmeye yönlendirebileceğinden, soruların cevaplanabilir olması ve mevcut bilgilerle tutarlı olması zorunluluğu vardır [18].

3.1.6. AHP'de temel ölçek kullanımı

Kişilerin bir soruna ilişkin bilgi düzeyleri arttıkça, söz konusu sorunun daha tutarlı bir şekilde modelini oluşturmaları beklenir. İkili karşılaştırmalar kişinin, soruna ilişkin olabildiğince bilgi kullanıp, tutarlılığını arttırmasına yardımcı olur. İkili karşılaştırmalar yapılırken yargılama için aşağıdaki çizelgede verilen ölçek kullanılır. Thomas L. Saaty tarafından ortaya atılan, "1-9 ölçeği" olarak adlandırılan ve daha sonra AHP çerçevesinde kullanılan göreli önceliklendirme ölçeği Çizelge 3.1'de verilmiştir. Bu çizelge sayesinde ikili karşılaştırmalar matrisi oluşturulmaktadır [22].

Çizelge 3.1. AHP'de kullanılan temel ölçek ve tanımları

	Tanım	Açıklama
1	Eşit önemli	İki seçenekte eşit derecede öneme sahip
3	Orta derecede önemli	Tecrübe ve yargı bir kriteri diğerine karşı biraz üstün kılmakta
5	Kuvvetli derecede önemli	Tecrübe ve yargı bir kriteri diğerine karşı oldukça üstün kılmakta
7	Çok kuvvetli derecede önemli	Bir kriter diğerine göre üstün sayılmıştır
9	Kesin önemli	Bir kriterin diğerinden üstün olduğunu gösteren kanıt çok büyük güvenilirliğe sahiptir
2,4,6,8	Ara değerler	Uzlaşma gerektiğinde kullanılmak üzere iki ardışık yargı arasındaki değerler

3.1.7. İkili karşılaştırmalar matrisi

İkili karşılaştırmaları elde etmek için göreceli veya mutlak ölçümler kullanılır. Bunlardan elde edilen bilgilere göre AHP'de yargılar bir matrise dönüştürülür. a_{ij}, i. özellik ile j. özelliğin ikili karşılaştırma değeri olarak gösterilecek olursa, genel olarak ikili karşılaştırma matrisi;

$$A=\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ . & .. & & . \\ a_{n1} & .. & & a_{mn} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & .. & a_{1n} \\ 1/a_{12} & . & & . \\ . & . & & . \\ 1/a_{1n} & . & & a_{mn} \end{pmatrix} \quad (3.1)$$

olur. a_{ji} ise, j. özellik ile i. özelliğin karşılaştırma değeridir. Bu değer a_{ij} değeri verilmişse; $a_{ji} = 1/a_{ij}$ eşitliğinden elde edilir. Bu özelliğe karşılık olma özelliği denir. Yukarıdaki ikili karşılaştırma matrisinin çözümünden elde edilecek öncelik veya

özdeğer vektörü $W=(w_1, w_2,..., w_n)$ ile gösterilir. w_i, öncelik veya özdeğer olarak tanımlanır. Bu değerlerden W^* matrisi elde edilir.

$$W^* = \begin{array}{c} \begin{matrix} A_1 & A_2 & & A_n \end{matrix} \\ \begin{pmatrix} w_1/w_1 & w_1/w_2 & \cdots\cdots\cdots & w_1/w_n \\ w_2/w_1 & w_2/w_2 & \cdots\cdots\cdots & w_2/w_n \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ w_n/w_1 & w_n/w_2 & \cdots\cdots\cdots & w_n/w_n \end{pmatrix} \end{array} \qquad (3.2)$$

Eğer sonuçlar tutarlı ise A ve W^* matrislerinin elemanları arasında çok büyük farkların olmaması gerekir.

İkili karşılaştırma matrisinin temel özellikleri [20]:

1. Temel ölçek olarak AHP'de 1-9 skalası kullanıldığı için A matrisinin öğeleri daima pozitif ve kare matris olacaktır. Yani ikili karşılaştırma matrisi pozitif değerlerden oluşmaktadır .

$$a_{ij}>0 \text{ , } i,j=1,2,...,n \qquad (3.3)$$

2. İkili karşılaştırma matrisi veya yargı matrisi eğer tam tutarlı ise aşağıdaki eşitliği sağlar .

$$a_{ij}.\, a_{jk} = a_{ik} \qquad i,j,k=1,2,...,n \qquad (3.4)$$

$$a_{ij}.\, a_{jk} = (w_i/w_j).\,(w_j/w_k) = w_i/w_k = a_{ik} \qquad i,j,k=1,2,...,n \qquad (3.5)$$

Bu özelliğin yani tam tutarlılığın göreceli karşılaştırmalarda elde edilmesi oldukça zordur. Bu nedenle AHP'de ağırlıkların veya öncelik vektörünün hesaplanmasında bazı farklı yöntemler kullanılmaktadır. Eğer A matrisi tam tutarlı ise öncelik veya ağırlık vektörlerini elde etmek oldukça kolaylaşmaktadır .

3. Eğer A matrisi tam tutarlı ise herhangi bir satırından matrisin diğer tüm öğeleri kolaylıkla elde edilebilir .

4. Toplam olarak C(n,2) kadar karşılaştırma yapılır. Örneğin 7 alternatifli bir durum için karar verici 21 ayrı ikili karşılaştırma yapacaktır .

5. Bu matrisin en büyük özdeğerine karşılık gelen özvektör matrisi AHP'de ağırlık veya öncelik vektörü olarak adlandırılır .

6. A matrisinin köşegen değerleri 1'e eşittir .

3.1.8. AHP'nin teorik temelleri

Saaty tarafından literatüre kazandırılan bu yöntem bir dizi aksiyom ve teoremlerden oluşmaktadır [20]:

Aksiyomlar

Aksiyom 1 (karşılık olma): Eğer bir a kriteri b kriterine göre x kez daha önemli ise, b kriteri de a'ya göre 1/x kez daha önemlidir. $a_{ij}=x$ ise $a_{ji}=1/x$'dir.

Aksiyom 2 (homojenlik): İkili karşılaştırmalarda a ve b kriterleri için biri diğerine göre ∞ kez üstün kabul edilemez. Yani $a_{ij}\neq\infty$ ($\forall$ i ve j'ler için) dir. Kullanılan ölçek 1-9 aralığında olduğu için a_{ij} değerleri de 1/9, 1/8,...,1,...,7,8,9 aralığında bir değer alacaktır.

Aksiyom 3 (bağımsızlık): Kriterler kendi aralarında ve alternatiflerden bağımsızdır.

Aksiyom 4 (beklenti): Bir karar problemi hiyerarşik yapıda sunulabilir.

Teoremler

AHP tekniği karar problemlerini çözerken aşağıdaki teoremleri izler:

Teorem 1: A matrisinin özdeğerleri λi (i=1,2,...,n) olarak gösterilsin. Bu özdeğerler aşağıdaki eşitliği sağlarlar.

$$\Sigma_{j,k=1}^{n} \lambda_i \lambda_k = 0 \qquad j \neq k \text{ için} \tag{3.6}$$

Teorem 2: $A=(a_{ij})$, $a_{ij} = a_{ji}^{-1}$ olmak üzere pozitif değerli ve n×n boyutlu bir kare matris olsun. A $\lambda_{max}=n$ ise tam tutarlıdır.

Teorem 3: ikili karşılaştırma matrisi tam tutarlı ise matrisin çeşitli derecelerden gücünü hesaplamak oldukça kolaydır. n, aktivite sayısını ve k'da istenilen kuvveti göstermek üzere; $A^k=n^{k-1}.A$ eşitliğinden elde edilir .

3.1.9. Öncelik vektörlerinin hesaplanması

Bu aşama, en büyük özdeğer ve bu özdeğere karşılık gelen özvektörün hesaplanmasını ve normalize edilmesini içermektedir. Bu amaçla kullanılan çeşitli yöntemler mevcuttur. Ancak literatürde en yaygın olarak kullanılan normalizasyon yönteminde her sütunun elemanları o sütunun toplamına bölünür. Elde edilen değerlerin satır toplamı alınıp ve bu toplam satırdaki eleman sayısına bölünür [4]. Bu şekilde her kriter için öncelik vektörleri bulunur.Bu prosedür, kriterlerin ikili karşılaştırma matrisine uygulanır. Kriterlerin öncelik vektörü, daha sonra her bir kritere göre alternatiflerin ikili karşılaştırma matrisine uygulanarak alternatiflerin öncelik vektörü elde edilir.

3.1.10. Tutarlılığın kontrolü

Genellikle karmaşık karar verme problemlerinin çözümünde bir miktar tutarsızlık söz

konusu olmaktadır. Karar vericinin kriterler arasında karşılaştırmaları yaparken tutarlı olup olmadığını görmek üzere her bir matris için Tutarlılık Oranı (TO, Eş 3.8) bulunur. Bulunan tutarlılık oranının 0,10 veya daha düşük olması yeterli görülmektedir [7].

Tutarlılık oranının hesaplanması;

1. İkili karşılaştırmalar matrisi ile bu matrise ait öncelik vektörü çarpılır. Elde edilen vektöre ağırlıklandırılmış toplam vektörü denir.
2. Elde edilen ağırlıklandırılmış toplam vektörünün her bir elemanı buna karşılık gelen öncelik vektörüne bölünür.
3. Adım 2 de elde edilen değerlerin ortalaması alınır ve buna maksimum özdeğer denir ve λ_{max} simgesi ile gösterilir.
4. Tutarlılık İndeksi = $(\lambda_{max} - n) / n-1$ (3.7)

n: karşılaştırılan eleman sayısı

5. Tutarlılık Oranı(TO) = Tutarlılık İndeksi(Tİ) / Rassallık İndeksi (Rİ) (3.8)

Rassallık İndeksi ikili karşılaştırma matrislerinin ortalama tutarlılık indeksini ifade eder. 1-15 boyutundaki matrisler için rassallık indeksi Çizelge 3.2'de gösterilmektedir [22].

Çizelge 3.2. Rasallık indeksi

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Rİ	0	0	0,58	0,9	1,12	1,24	1,32	1,41	1,45	1,49	1,51	1,48	1,56	1,57	1,59

Karar teorisinin en önemli temel taşlarından birisi olan tutarlılık için hiçbir ölçüm türü kesin olarak garanti veremez. Bu yüzden bir miktar tutarsızlığa izin verilmelidir. Tutarsızlık aşırılıktan kaynaklanmaktadır. Tutarsızlığın %10 olarak alınması, karşılaştırmaları yapılan öğelerin özüne zarar verilmeden, ölçümlerde bir takım varyasyonların denenmesine olanak sağlayacaktır.

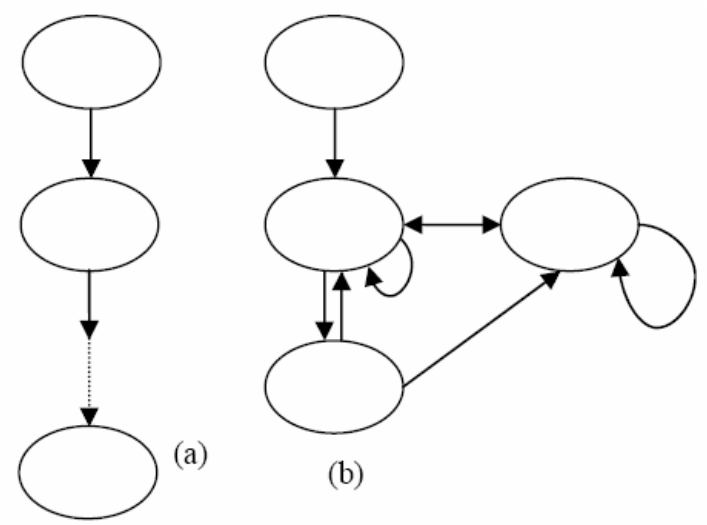
3.1.11. Nihai karar

AHP'nin son aşaması karar probleminin çözümlenmesi aşamasıdır. Bu aşamada problemin ana hedefinin gerçekleştirilmesinde karar alternatiflerinin sıralaması olarak hizmet edecek bir karma öncelikler vektörü oluşturulur. Bu vektörü oluşturmak için her değişken için belirlenen öncelik vektörlerinin ağırlıklı ortalaması alınır . Elde edilen nihai öncelikler karar alternatif puanları olarak da adlandırılabilir. Karar verici elde ettiği ağırlıklara göre kararını verir [23].

3.2. Analitik Network Prosesi (ANP)

AHP'nin en önemli varsayımlarından biri aynı seviyede bulunan faktörlerin birbirinden bağımsız olması ve faktörlerin birbirine olan etkilerinin dikkate alınmamasıdır. Oysa gerçek hayatta karar verme problemlerini etkileyen birçok faktör birbiriyle etkileşim halinde bulunmakta ve en iyi kararın verilmesi faktörler arasındaki bu ilişkilerin dikkate alınmasını gerektirmektedir. Karar verme sürecinde faktörler arasındaki ilişkileri dikkate alan ve problemin tek bir yöne bağlı kalarak modelleme zorunluluğunu ortadan kaldıran yöntem yine Thomas L. Saaty tarafından geliştirilen Analitik Network Prosesi (ANP) yöntemidir [24]. Verilen kararların daha gerçekçi olması için karar modellerinin somut ve soyut, niteliksel ve niceliksel faktörleri de içermesi ve ölçmesi gerekir.

AHP, ANP'nin özel bir halidir. AHP birimlerin tek yönlü ilişkilerine, ANP ise karar seviyeleri ve özellikler arasında daha karmaşık ilişkilere izin verir. Bu şekilde hiyerarşik olarak modellenemeyen karmaşık problemlerin kolayca modellenmesini sağlar. Şekil 3.4'te bir hiyerarşi ve bir network yapısı gösterilmiştir.



Şekil 3.4. Bir network ile bir hiyerarşi arasındaki yapısal fark: (a) Hiyerarşi (b) Network

3.2.1. Bağımlılıklar

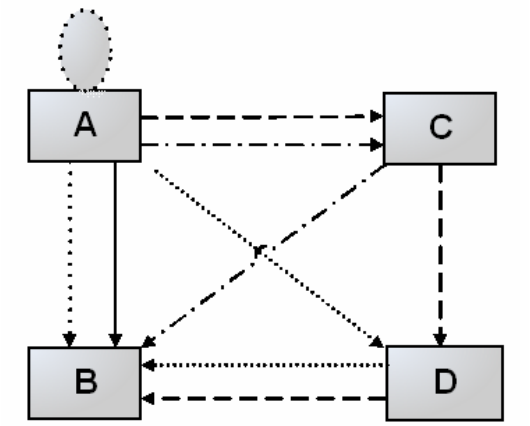
Dış bağımlılık (Outer dependence): Kümeler arasındaki bağımlılığa denilmektedir. Bir başka deyişle, dış bağımlılık bir kümedeki elemanlar ile diğer kümelerin elemanları arasındaki etkileşimdir [25].

İç bağımlılık (Inner dependence): Bir kümedeki bir elemanın aynı kümedeki başka bir eleman üzerindeki etkisidir [25].

ANP'de ikili karşılaştırmalar matrislerinden elde edilen öncelik vektörleri, AHP'de olduğu gibi doğrusal bir biçimde birleştirilmez. Bir kümedeki elemanların diğer kümelerdeki elemanlara etkisini *(dış bağımlılık)* ya da aynı kümedeki diğer elemanlara etkisini *(iç bağımlılık)* göstermek için bu vektörler bir matrise kolon olarak yerleştirilirler. Bir kümedeki elemanların hepsinin, başka bir kümedeki bir elemanı etkilemesi zorunluluğu yoktur ve etkisi olmayan bu elemanların katkıları sıfır değeri verilerek gösterilebilir.

3.2.2. Etki matrisi (The impact matrix)

Geri beslemeli bir modelde elemanlar arasında *"Önceliklerin birleştirilmesi"* kavramı özel bir dikkat gerektirir. Modeldeki elemanlar tek bir yol ile değil, birden fazla yol boyunca etkileşim halinde olabilirler.



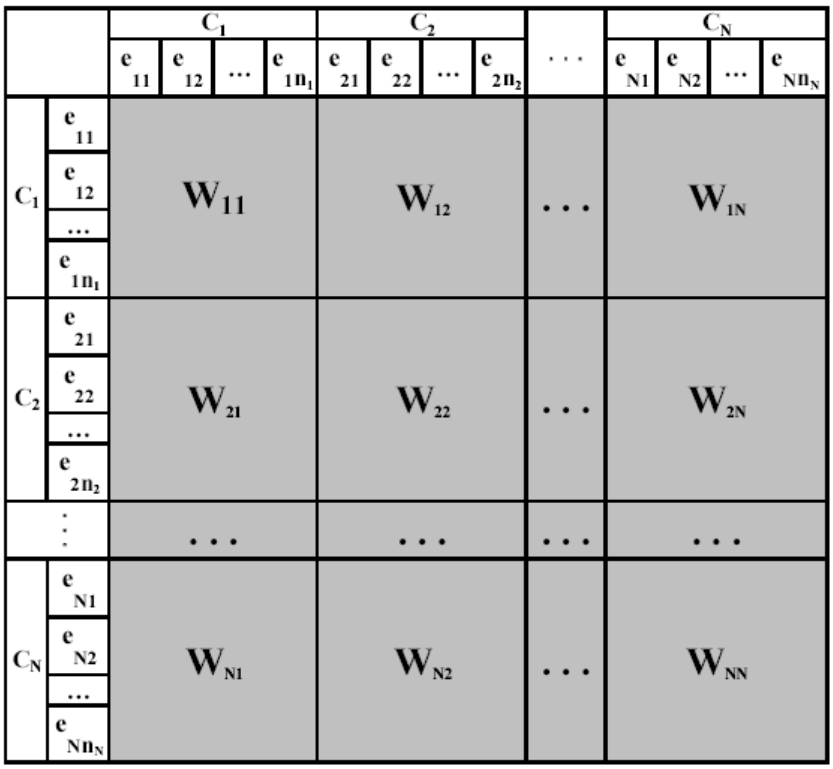
Şekil 3.5. A elemanının B elemanı üzerinde farklı yollardan etkileri

Önceliklerin ölçümünün anlamlı olması için, olası bütün yolların aynı şekilde düşünülmesi gerekir. Modeldeki herhangi bir elemanın bir diğerine göre önceliği, onları birbirine bağlayan yollar ve döngüler üzerinden, birden fazla biçimde ölçülebilir. Herhangi bir A elemanının B elemanı üzerindeki toplam etkisini düşünelim. A elemanının B elemanı üzerindeki doğrudan etkisi (birinci dereceden etkisi de denilebilir) Şekil 3.5' te düz çizgi ile görülmektedir. Bu şekilde birinci dereceden bütün etkiler, *etki matrisinden* (süpermatris de bir tür etki matrisidir) görülebilir. A elemanının B elemanı üzerindeki etkisi dolaylı olarak C elemanı üzerinden de olabilir. Şekil 3.5' te noktalı çizgilerle gösterilen bu etkinin, toplam etkiye katkısı A elemanının C elemanına etkisi ile C elemanının B elemanına etkisinin çarpımından elde edilebilir. A elemanının B elemanı üzerindeki ikinci dereceden dolaylı tüm etkilerinin toplamı, *etki matrisi*nin karesinden elde edilebilir. Şekil 3.5' te kesikli çizgilerle gösterildiği gibi, A elemanının B elemanı üzerinde üçüncü dereceden dolaylı etkisi de vardır. Bu etkinin toplam etkiye katkısı; A elemanının C elemanına etkisi, C elemanının D elemanına etkisi ve D elemanının B elemanına etkisinin çarpımlarından elde edilebilir. İkinci dereceden

dolaylı etkilerde olduğu gibi, üçüncü dereceden dolaylı etkilerin tamamı da *etki matrisi*nin küpünden elde edilebilir ve bu dördüncü, beşinci, altıncı dereceden dolaylı etkiler hesaplanarak devam eder [26].

3.2.3. Süpermatris

Bütün ikili karşılaştırmalar matrislerinden elde edilen öncelik vektörlerinden ve sıfır vektörlerinden oluşan büyük matris yardımıyla, sonuç önceliklerin anlamlı limitleri elde edilebilir. Bunun için bu matrisin stokastik (her bir kolonun toplamı bir olacak şekilde) bir matris olması gerekmektedir. Bu büyük matriste elemanlar, dikey olarak matrisin sol tarafında ve yatay olarak matrisin üst tarafında gösterilirler. *Süpermatris* olarak adlandırılan bu matrisin stokastik olmasını sağlamak için, sisteme ait ayrı bir *kontrol hiyerarşisi*nde gösterilen özellikler bakımından, sol tarafta bulunan kümelerin üst tarafta bulunan her bir küme üzerindeki etkilerine göre birbirleri ile karşılaştırılmaları gerekir. Bunun sonucunda elde edilen *kümelerin öncelikleri* daha sonra kolon vektörlerini ağırlıklandırmak için kullanılır. Kolon vektörlerinden oluşan her bir blok (üst taraftaki bir kümenin elemanları ile sol taraftaki bir kümenin elemanlarını kapsar), süpermatrisin bir elemanıdır. Bloktaki kolon vektörlerinin hepsi üst taraftaki bir küme açısından sol taraftaki ilgili kümenin önceliği ile çarpılır. Süreç sol taraftaki bütün kümelerin üst taraftaki her bir küme açısından öncelik vektörlerini elde ederek tekrarlanır. Böylece, süpermatrisin üst tarafındaki bir kümenin elemanları üzerindeki etkileri gösteren kolonların her birinin toplamı bir olacaktır. Artık süpermatris kolon stokastik olmuştur.



Şekil 3.6. Süpermatrisin genel yapısı [25]

3.2.4. Limit öncelikleri (The limiting priorities)

Asıl istenilen, her bir elemanın diğer bütün elemanlar üzerindeki etkilerinin *limit öncelikleri*nin bulunmasıdır.

$$\lim_{k \to \infty} W^k \qquad (3.9)$$

Ağırlıklandırılmış süpermatris elde edildikten sonra matrisin kuvvetleri alınarak yakınsak öncelikler bulunur(Eş.3.9). Kuvvet alma işlemine satırdaki sayılar eşitleninceye kadar devam edilir. Böylelikle matrisin limit denge dağılımı kısaca 'Limit süpermatris' elde edilir. Limit süpermatris Ecnet, Super Decisions gibi analitik network yapısını çözen programlarla ya da Mathematica, Matlab, Excel gibi matematiksel çözüm yapabilen programlarla hesaplanabilir.

3.2.5. Kontrol hiyerarşisi (The control hierarchy)

ANP analizi için kritik olan kontrol hiyerarşisi, network düzenindeki ilişki çeşitlerini karşılaştırmak için önemlidir. iki çeşit kontrol kriteri vardır. Eğer yapı hiyerarşikse, bir kontrol kriteri yapıya hiyerarşinin amacı olarak bağlanabilir. Bu durumda kontrol kriteri bir karşılaştırma, bağlantı kriteri olarak nitelendirilir. Aksi durumlarda kontrol kriterleri yapıya direkt olarak bağlanmaz. Fakat şebeke içinde karşılaştırmalara neden olur. Bu durumda, kontrol kriteri karşılaştırma, "neden olma" kriteri olarak nitelendirilir.

Kontrol hiyerarşisinde ağırlıklandırma işlemi karşılaşılan sorunlardan biridir. Grupların ağırlıkları, o gruplardan etkilenen gruba karşılık gelen süpermatris bloklarının ağırlıklarını hesaplamada kullanılır. Her süpermatristeki yakınsak öncelikler kendilerine karşılık gelen alt kriterin öncelikleriyle çarpılarak ağırlıklandırılırlar. Şayet bir öğenin veya grubun hiç girdisi yoksa karşılık gelen öncelik vektörüne sıfır girilir [27].

ANP'nin adımlarını özetleyecek olursak [7];

1. Problemin modellenip şebekenin oluşturulması,
2. Kriter ve alt kriterler arasındaki ikili karşılaştırmalar matrislerinin oluşturulması,
3. Göreli öncelik vektörlerinden oluşturulan alt matrislerin süpermatristeki yerlerine yerleştirilmesi,
4. Süpermatrisin sütun toplamlarını bir yaparak stokastik matrisin yani ağırlıklandırılmış süpermatrisin elde edilmesi,
5. Ağırlıklandırılmış süpermatrisin kuvveti alınarak limit süpermatrisin elde edilmesi

4. YAZILIM GELİŞTİRME VE BU ALANDAKİ YAKLAŞIMLAR

4.1. Programlama Dillerinin Seviyelerine Göre Sınıflandırması ve Gelişimi

Bir programlama dilinin seviyesi o programlama dilinin insan algısına olan yakınlığının derecesi ile doğru orantılıdır. Bir programlama dili insan algılasına ne kadar yakınsa o kadar yüksek seviyeli demektir. Yine bir programlama dili bilgisayarın elektronik yapısına ve çalışma biçimine ne kadar yakınsa o kadar düşük seviyeli demektir. Programcılar için yüksek seviyeli dillerle çalışmak kolaydır. Bu dillerde yalnızca nelerin yapılacağı programa bildirilir, nasıl yapılacağı bildirilmez. Genel olarak programlama dilinin seviyesi arttıkça, o dilin öğrenilmesi ve o dilde program yazılması kolaylaşır.

Genel olarak bilgisayar dilleri üç ana seviyeye göre sınıflandırılır: makine dili, sembolik makine dili(assembly dili) ve yüksek seviyeli dil.

Bir bilgisayar yalnızca kendi makine dilini doğrudan anlayabilir. Makine dili bilgisayarın doğal dilidir. Bilgisayarların geliştirilmesiyle birlikte onlara iş yaptırmak için kullanılan ilk diller de makine dilleri olmuştur. Bu yüzden makine dillerine birinci kuşak diller denilmektedir.

Makine dilinin programlarda kullanılmasında iki temel problemle karşılaşılmaktadır. Birincisi makine dili taşınabilir değildir. Diğeri ise, makine dilinde kod yazmanın çok zaman alıcı ve uğraştırıcı olmasının yanı sıra yazılan programı algılamanın da o denli zor olmasıdır.

Yazılımın ve donanımın tarihsel gelişimi içerisinde makine dilinden, insan algılamasına çok yakın yüksek seviyeli dillere kadar uzanan bir süreç söz konusudur. Bu tarihsel süreci ana hatlarıyla aşağıdaki gibi gözden geçirmek mümkündür;

Makine dili kullanımının getirdiği problemleri ortadan kaldırmaya yönelik çalışmalar 1950 li yılların başlarında yoğunlaştı. Bu yıllarda makine dilleri bilgisayarın çok

sınırlı olan belleğine yükleniyor ve programlar böyle çalıştırılıyordu. İlk önce makine dilinin algılanma ve anlaşılma zorluğunu kısmen de olsa ortadan kaldıran bir adım atıldı. Sembolik makine dilleri(assembly languages) geliştirildi. Sembolik makine dilleri yalnızca 1 ve 0 dan oluşan makine dilleri yerine İngilizce bazı kısaltılmış sözcüklerden oluşuyordu. Sembolik makine dillerinin kullanımı kısa sürede yaygınlaştı. Fakat sembolik makine dillerinin de çok önemli bir dezavantajı vardı. Bu dillerde yazılan programlar makine dilinde yazılan programlar gibi bilgisayarın belleğine yükleniyor ancak programın çalıştırılma aşamasında yorumlayıcı bir program yardımıyla sembolik dilin komutları, bilgisayar tarafından komut komut makine diline çevriliyor ve makine diline çevrilmiş komutları icra ediyordu. Bu şekilde çalıştırılan programların hızı oldukça yavaşlıyordu.

O zamanlar Grace Hopper isimli bayan tarafından ortaya atılan fikir şu idi: Her defasında yazılan kodun, çalıştırılması sırasında makine diline çevrileceğine, geliştirilecek bir başka program sembolik dilde yazılan kodu bir kez makine diline çevirsin ve artık program ne zaman çalıştırılmak istenirse, bilgisayar yorumlama olmaksızın yalnızca makine kodunu çalıştırsın. Grace Hopper'ın buluşuna "compiler-derleyici" ismi verildi. Artık programcılar sembolik makine dili programlarını kullanıyor, yazdıkları programlar derleyici tarafından makine koduna dönüştürülüyor ve makine kodu eski hızından birşey kaybetmeksizin tam hızla çalışıyordu. Sembolik makine dilleri ikinci kuşak diller olarak tarihte yerini aldı.

Tarihsel süreç içinde sembolik makine dillerinden sonra geliştirilmiş ve daha yüksek seviyeli diller üçüncü kuşak diller sayılmaktadır. 1950'lerin başlarında geliştirilen yüksek seviyeli diller, bazı işleri programlamanın dışında tutarak sembolik makine dillerinin can sıkıcı özelliklerinin üstesinden gelmiş ve programcıya elindeki problemi çözmeye daha çok konsantre olma fırsatını vermiştir. Bugüne kadar yüzlerce yüksek seviyeli dil geliştirilmiştir. Bunlardan popüler olanları: COBOL, Pascal, FORTRAN, BASIC, LISP, ADA ve C/C++ [28]. Bu dillerin hepsi algoritmik dillerdir.

Çizelge 4.1'de aralarındaki farkı daha somut bir şekilde göstermek için makine diline, sembolik makine diline ve yüksek seviyeli dilere ait birer örnek gösterilmiştir [28].

Çizelge 4.1. Bilgisayar dillerinin sınıflandırılması

(a) Makine dili	**(b)**Sembolik makine dili	**(c)** Yüksek seviyeli dil
01001100	mov bx, offset value	x=2;
11101001	mov ax, [bx]	if (x<=y)
10101010	add ax, 5	x = x + 1;
10001110	add bx, 2	else
00001111	add ax, [bx]	x = x - 1;

FORTRAN (Formula translator), IBM tarafından 1954 ile 1957 yılları arasında bilimsel uygulamalarda ve mühendislik uygulamalarında kullanılan matematik hesaplamalarını yapmak için geliştirilmiştir. Fortran, özellikle mühendislik uygulamalarında hala yaygın bir biçimde kullanılmaktadır [29].

COBOL (Common Bussiness Oriented Language), 1959'da bilgisayar üreticileri, devlet ve endüstriyel bilgisayar kullanıcıları tarafından geliştirilmiştir. COBOL, büyük verilerin kullanılmasını gerektiren ticari uygulamalarda kullanılmaktadır. Birçok iş yazılımı COBOL ile programlanmaktadır [29].

Yapısal Programlama

1960'larda yazılım geliştirme çabaları ciddi zorluklarla karşılaştı. Yazılım takvimleri genellikle geç kalıyordu, maliyetler yatırımları aşıyordu ve bitirilmiş projeler yeterince iyi değildi. İnsanlar, yazılım geliştirmenin düşünülenden çok daha zor bir iş olduğunun farkına vardılar. 1960'lardaki araştırma çalışmaları, *yapısal programlama*nın ortaya çıkmasına sebep olmuştu. Yapısal programlama ile programları daha açık, daha doğru ve değiştirilmesi daha kolay yazabilmek için bir disiplin oluşturulmuştu [29].

1960'lardaki araştırmanın en önemli sonuçlarından birisi, PASCAL programlama dilinin 1971 yılında Profesör Niklaus Wirth tarafından geliştirilmesi olmuştur. PASCAL ismi 17. yüzyılın matematikçi ve filozoflarından Blais PASCAL'dan geliyordu ve yapısal programlamayı akademik çevrelerde öğretmek amacıyla

tasarlanmıştı. Pascal, hızlı bir biçimde çoğu üniversitelerde programlama dillerine girişte tercih edilir hale gelmişti fakat bu dil ticaret ve endüstriyel uygulamalar ile hükümetin istediği uygulamaları gerçekleştirmek için çok önemli bazı özelliklerden yoksundu. Bu yüzden de bu çevrelerde çok geniş bir kullanım alanı bulamadı [29].

Her geçen gün programların daha karmaşık bir hal alması, program kodunun kurumsal uygulama projelerinde onbinlerce satırı bulması ve yazılım geliştirme maliyetinin çok arttığını gören bilim adamları, programcılara yeni bir yaklaşımın kullanılabileceğini gösterdiler. Bu yaklaşıma "Nesne Yönelimli Programlama" ismi verilmiştir.

Nesne Yönelimlilik (Object Orientation)

Nesne yönelimlilik de yapısallık gibi bir programlama tekniğidir. Bu teknik kaynak kodların çok büyümesi sonucunda ortaya çıkan gereksinim yüzünden geliştirilmiştir. C dilinin geliştirildiği yıllarda, akla gelebilecek en büyük programlar ancak onbin satırlar mertebesindeydi, ancak kullanıcıların bilgisayar programlarından beklentilerinin artması ve grafik arayüzünün artık etkin olarak kullanılmasıyla, bilgisayar programlarının boyutu çok büyümüştür. Nesne yönelimli programlama tekniği, herşeyden önce büyük programların yazılması için tasarlanmış bir tekniktir.

Nesne yönelimli programlama tekniğinin yaygın olarak kullanılmaya başlanmasıyla birlikte bir çok programlama dilinin bünyesine bu tekniğin uygulanmasını kolaylaştırıcı araçlar eklenek, yeni versiyonları oluşturulmuştur. Örneğin C'nin nesne yönelimli programlama tekniğini uygulayabilmek için 1980'lerde Bjarne Stroustrup tarafından geliştirilmiş haline C++ denmektedir [29]. C++ dili, C dili baz alınıp geliştirilmiş yeni bir programlama dilidir. Bazı programlama dilleri ise doğrudan nesne yönelimli programlama tekniğini destekleyecek şekilde tasarlanarak geliştirilmiştir.

Nesne yönelimli programlamanın sunduğu olanakların yeterli şekilde değerlendirilebilmesi için önce geleneksel yordamsal (procedural) programlama yaklaşımından kaynaklanan sorunların ele alınması gerekir.

4.2. Yordamsal (Procedural) Programlama

Yordamsal programlamada yazılımlar birbirini çağıran bir dizi yordam (procedure) ve işlevler toplulugu olarak geliştirilir. Her yordam ve işlev kendi yerel verisini, yine kendi yerel degişkenlerinde tutar. Paylaşılması gereken veriler yordam çağırma komutlarında parametre olarak yordamdan yordama geçirilir. Parametrelere sığmayacak büyük veriler ise genel (global) degişkenler içerisinde herkesin kullanımına açılır.

Yordamsal Programlamanın Zayıf Yanları

Yordamsal Programlamanın zayıf yanları aşağıda sıralanmıştır [30]:

• Genel kullanıma açılan veriler tümüyle korumasız kalır.

• Verinin kullanım amacı veri üzerinde yapılabilecek işlemleri hiçbir biçimde sınırlamaz.

• Veriyi tutan degişken genel kullanıma açıldıktan sonra, o değişken türünün desteklediği her türlü işlem veriye uygulanabilir.

• Amaç dışı kullanımdan kaynaklanan yanlışlıklar ortaya çıktıktan sonra, yanlışlığa neden olan program kesiminin saptanması zordur. Bunun için sözkonusu veriye erişen tüm yordamların tek tek incelenmesi gerekir.

• Yordamsal programlamada verinin saklanma biçimi, veriye erişim ve verinin işlenme biçimini doğrudan etkiler. Aynı verinin sayısal bir degişken yerine karakter türü bir değişkende tutulmasına karar verildiğinde, tek boyutlu dizi yerine iki boyutlu

dizi kullanıldığında vs. sözkonusu veriye erişen tüm yordamların elden geçirilmesi gerekir. Bu durumda yöntem değişikliğinden etkilenen tüm kod kesimleri eksiksiz saptanmalı ve gereken güncellemeler hatasız olarak yapılmalıdır. Daha sonra programlar yeniden sınanır, derlenir ve kullanıcılara dağıtılır. Bu işlemler sırasında bazı kod kesimlerinin atılması ve yenilerinin eklenmesi de gerekebilir. Sonuç olarak, her aşamada sisteme yeni yanlışların sızması ya da dolaylı ilişkilerden ötürü ancak belli süre sonunda saptanabilecektir ve bulunması daha zor hataların ortaya çıkması olasılığı yüksektir.

• Yordamsal programlamada eldeki kodun yeniden kullanımına dönük altyapı zayıftır. Farklı işletmelerdeki belli işlevlerdeki küçük farklılıklar geliştirilen programın özel durumlar için özel kesimler içermesine ya da programın farklı durumlar için farklı kopyalarının üretilmesine neden olur. İlk seçenekte varolan genel amaçlı parametrik programların tasarımı, gerçekleştirilmesi ve sınanması daha zor, pahalı ve zaman alıcıdır. Bunların anlaşılması ve bakımı da zordur. İkinci seçenekte ise bir kopyada saptanan hatanın diğer kopyalarda düzeltilmesi, güncellemelerin yapılması çok zahmetli ve tekrarlıdır. Uygulamada bir kopyada yapılan değişikliğin her kopyaya yansıtılamadığı ve her kopyanın başarısının farklılaştığı görülmektedir.

Yordamsal programlamadaki önemli sorunlardan biri de programcıların yarattığı program birimlerinin, gerçek dünyayı tam olarak yansıtamamasıdır. Bu sebepten, bu birimler yeniden kullanılabilir değildir. Programcıların, baştan başladıktan sonra diğer kodlara yakın kodlar yazmaları oldukça sık görülen bir durumdur. Bu durum, zaman ve yatırım maliyetlerini artırır. Çünkü her seferinde tekerlek yeniden icat edilir [29].

Sonuç olarak yordamsal programlamadaki zorluklar şöyle özetlenebilir:

• Hataların saptanma ve düzeltilmesindeki zorluk.
• Programın kullanıcı gereksinmelerini ve yenilik isteklerini tam karşılayacak biçimde hızla değiştirilebilir olmaması ve her yapılan ek ya da degişikliğin programın daha önce çalışan ve değiştirilmeyen kesimlerinde bile hataların oluşmasına yol açabilmesi.

• Eldeki sınanmış kod kesimlerinin ciddi değişiklikler yapılmaksızın yeni gereksinmelerin karşılanmasında kolayca kullanılamaması.

4.3. Nesne Yönelimli Programlama (Object-Oriented Programming)

Nesne teknolojisi, belirli uygulama alanlarında büyük ve odaklanmış anlamlı yazılım birimleri oluşturmamıza yardımcı olan paketleme şemasıdır. Her isim, bir nesne olarak gösterilebilir. Nesnelerle dolu bir dünyada yaşıyoruz. Etrafımıza baktığımızda bir çok nesne görmekteyiz; arabalar, uçaklar, insanlar, hayvanlar, binalar, trafik ışıkları, anahtarlar ve benzerleri. Nesne yönelimli dillerden önce, programlama dilleri (FORTRAN, PASCAL, BASIC ve C gibi) nesneler yerine eylemlere odaklanmıştı. Nesnelerle dolu bir dünyada yaşayan programcılar, bilgisayarların başına geçince eylemlerle uğraşıyorlardı [29].

Numune değişimi, program yazmayı hantal hale getiriyordu. Şimdi ise Java, C++ ve bir çok diğer nesne yönelimli programlama dilleri sayesinde programcılar, normal yaşamlarında olduğu gibi bilgisayarlarının karşısında da nesnelerle uğraşmaya devam ederler. Bu, onların programlarını dünyayı gördükleri biçimde yazmaları anlamına gelir. Bu, yordamsal programlamadan daha doğal bir yoldur ve verimliliğin artmasına önemli katkılarda bulunmuştur [29].

Nesne yönelimli programlamada yazılımlar birbirleriyle iletişim kurabilen nesneler topluluğu olarak tasarlanır ve gerçekleştirilirler. Kod işletimi nesnelerin içinde yapılır ve her nesne bir diğer nesneye ileti göndererek ondan hizmet alabilir. Bu nedenle nesne yönelimli programlama "nesnelere ileti gönderme yoluyla programlama" olarak da isimlendirilir.

Nesneler *metodlar* (methods) ve *nitelikler*'den (attributes) oluşur. Nitelikler, nesnelerin sahip oldukları *verilere*, metodlar ise bunlar üzerinde yapılabilecek *işlemlere* karşılık gelir. Bir başka deyişle nesne, kendisini işleyecek kod kesimini kendisi ile birlikte tanımlayan ve taşıyan ve kendi tanımladığı biçimden daha farklı

amaçlarla kullanılamayan veri türü olarak yorumlanabilir. Nesnelerin genel özellikleri aşağıda tanımlanmıştır [30]:

• Nesneler sınıflar(class)'lardan oluşturulur. Sınıf, nesnenin oluşturulmasını sağlayan bilgileri içeren yapının adıdır. Bu nedenle sınıf, bir nesne şablonu olarak tanımlanabilir. Bir sınıftan türetilen tüm nesneler aynı şekilde davranır; bir diğer deyişle, hepsi aynı metot çağrılarına cevap verirler.

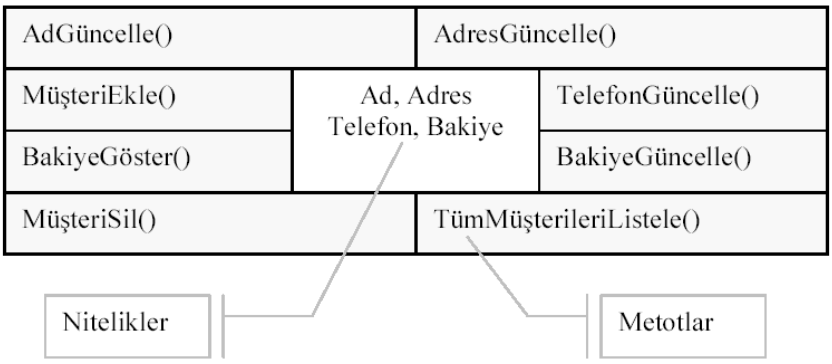
• Nesneler kalıcı ve geçici olabilirler. Nesneler ana bellekte oluşturulan ve ana bellekte kullanılabilen yapılardır. Dolayısıyla, önlem alınmazsa içinde yer aldıkları program sonlandırıldığında kaybolurlar.

• Nesnelerin, niteliklerinde tutulan değerlerle tanımlanan durumları (state) vardır. Bir başka deyişle, aynı sınıfa bağlı nesneleri birbirinden ayıran şey işleyiş biçimleri ya da yetenekleri değil, içlerinde tuttukları verilerin değerleridir.

• Nesnelerin yaşam döngüleri vardır. Bu yaşam döngüsü içinde oluşturulur, kullanılır ve yok edilirler.

• Nesneler dış dünyada arayüzleri (interface) ile bilinirler ve kullanılırlar. Bir nesnenin arayüzü, o nesnenin dış dünyanın kullanımına açtığı metotların listesinden oluşur. Bir diğer deyişle, arayüzler nesnelerin hangi hizmetleri sağladığını belirtir, ancak bunları nasıl yapacağı konusunda bilgi içermezler.

Aşağıda (Şekil 4.1) bir müşterinin ad, soyad, telefon ve bakiye bilgilerini tutan bir sınıf tanımı görülmektedir. Müşterinin bilgilerini içeren değişkenlere dış dünyadan erişim mümkün değildir. Bu bilgilere erişmek, okumak ya da değiştirmek isteyen bir program, bu sınıfın metotlarından birini - *Bakiye göster () ya da Adres Güncelle ()* gibi kullanmak zorundadır.



Şekil 4.1. Sınıf (class) örneği (Müşteri)[30]

Yazılımlar giderek daha fazla kullanıcı tarafından kullanıldıkça, yeni ihtiyaç ve isteklerin ortaya çıkması kaçınılmazdır. Öte yandan her iyileştirme, eklenti ve değişiklik isteminin yazılımın tasarımını az ya da çok etkilediği ve ek çaba gerektirdiği açıktır. Dolayısıyla her yazılımcının değişiklik istemlerinin ana tasarım üzerindeki sarsıcı etkilerini olabildiğince en aza indirgeyerek bu istekleri karşılamasını sağlayacak araçlara ihtiyacı vardır [30].

Nesne yönelimli programlama tekniği, diğer yaklaşımlara nazaran, yazılım geliştiren insanlara büyük avantajlar sağlamaktadır. Birincisi karmaşık yazılım projelerinin üretilmesini ve bakımını kolaylaştırıyor olmasıdır. Diğeri ise program kodunun tekrar kullanılabilmesine (code-reusability) olanak sağlamasıdır. Bu noktada program kodunun tekrar kullanılabilmesi profesyonel yazılım şirketlerinin maliyetlerini azaltmıştır. Dolayısı ile programların lisans ücretleri düşmüş ve sektörün sürekli olarak canlı kalmasına ve rekabet içinde gelişmesine yardımcı olmuştur. Bazı kaynaklar, nesne yönelimli programlamada esas faydanın yeniden kullanım olmadığını belirtmiştir. Bunun yerine, nesneye dayalı programlamanın daha anlaşılır ve daha iyi organize olmasının, incelenmesinin, değiştirilmesinin ve hataların ayıklanmasının daha kolay olmasının bu teknolojinin en önemli özellikleri olduğunu belirtmişlerdir. Bunlar, gerçekten de önemlidir. Çünkü yazılım maliyetlerinin %80'nin programlarının geliştirilmesinde ve iyileştirmelerin yapılmasında harcandığı belirlenmiştir [29].

4.4. Yazılım : Dumansız Endüstri

BT ve onun alt dalı olan yazılım "jenerik/doğurgan" bir teknolojidir. Kendisi dahil hemen her sektörde verimliliği ve etkinliği artırmak, kaliteyi yükseltmek amacıyla kullanılmaktadır. Bu nedenle üretimden sağlığa, eğitimden finansa kadar çok geniş bir kullanım alanı bulunmaktadır ve vazgeçilmez görülmektedir. Zira kullanan sektörlere ve kurumlara rekabetçi bir avantaj sağlamaktadır. Teknolojilerdeki gelişmelere paralel olarak kullanım alanları giderek çoğalmaktadır. Yakın gelecekte bilgi teknolojilerini ve dolayısı ile yazılımı kullanmayan hiçbir alan kalmayacaktır. Kullanmayanlar da zaten yok olacaktır [1].

Yazılım, dumansız endüstridir. Yazılım, sürdürülebilir kalkınma modelleri içinde yer alabilecek en uygun sektördür. Yazılım, çevre ile dost bir endüstridir. Herhangi bir girdi gerektirmemektedir. Çevreyi kirleten bir üretim süreci bulunmamaktadır. Çıktısı elektronik ortamda saklanabilen ve taşıma sorunu olmayan yumuşak bir üründür. Hiçbir doğal kaynağı tüketmediği, yok etmediği ve kirletmediği için kalkınma aracı olarak da öncelikle tercih edilen bir sektör olarak kabul edilmektedir [1].

4.5. Yazılım Sektörünün Yapısal Farklılıkları ve Gelişimine Katkı Sağlayacağı Düşünülen Görüşler

Yazılım Sektörü, diğer sektörlerden çok temel farklılıklar içermektedir. Bunlar kısaca şöyle özetlenebilir [1]:

- Yazılım diğer sanayi ürünlerinden farklıdır: Yer kaplamaz ve taşınması kolaydır.
- Yatırım sermayesi azdır. Bir ofis ile çalışanlara sağlanacak bilgisayar ve iletişim altyapısı, bir yazılım evinin çalışır duruma gelmesi açısından yeterlidir.
- Üretim süreci diğer sanayi ürünlerinden farklıdır: Bir kez üretilir, ama binlerce kez satılabilir.
- İthalata dayalı girdisi yoktur: Döviz götürmez, aksine döviz getirir.

- Enerji tüketimi dikkate alınmayacak kadar azdır. Yazılım sektörü için yeni enerji kaynakları yaratmak gerekmez.
- Katma değeri yüksek bir üründür ve tümüyle yurt içi kaynaklarla üretilebilir.

Bilgi ve iletişim teknolojileri toplumun tüm kesimlerine yayıldığı ve bir "ağ etkisi" oluşturduğu zaman bu teknolojilerin sağladığı katkı daha hızlı ve somut olarak ortaya çıkmaktadır. Örneğin, 1990'lı yılların ikinci yarısından itibaren, bilgi ve iletişim teknolojilerinin işgücü verimliliği artışı üzerindeki katkısının ABD için yüzde 60, Avrupa Birliği için yüzde 40 seviyelerinde olduğu tahmin edilmektedir. Aynı dönemde, Avrupa Birliğinde ekonomik büyümenin yüzde 25'i bilgi ve iletişim teknolojilerinden kaynaklanmıştır [31].

Bilgi ve iletişim teknolojileri ve gelişen küresel ekonomi, ülkelere kalkınma ve uluslararası rekabet yarışında büyük fırsatlar sunmakla beraber, yeni tehditleri de beraberinde getirmektedir. Teknolojik devrimlerin yaşandığı dönemlerde fiziki ve beşeri sermayenin bir kısmı ekonomik önemini kaybettiğinden teknolojik ve ekonomik açıdan ileri ülkeler mevcut avantajlarını yitirebilmektedir. Bu dönemler, iyi değerlendirildiği takdirde geriden gelen ülkeler açısından önlerindeki ülkeleri yakalayıp geçmek için önemli bir fırsat ortaya çıkarmaktadır. Öte yandan, gerideki ülkeler bu dönemi iyi değerlendiremedikleri takdirde hızlı bir şekilde bulundukları pozisyondan daha da geriye itilebilirler. Bu nedenle, ülkeler arası rekabet dengelerinin yeniden şekillendiği böyle dönemlerde, doğru politikaların belirlenerek hızla doğru adımların atılması her zamankinden daha önemli hale gelmektedir [31].

Devletin yazılım sektörünün gelişmesi için bir şeyler yapması, yeni bir bakış açısı, yeni bir vizyon edinmesi gerekmektedir. Yazılım sektörünün içinde bulunduğu sorunlar bellidir. Sektörün gelişmesi için bu sorunların çözümüne yönelik düzenlemeler yapılması gerekmektedir. Yazılım sektörünün gelişimine katkı sağlayacağını düşünülen bazı görüş ve öneriler [1]:

- Yazılım üretimi ülkenin stratejik destek vereceği konular arasına alınmalıdır. Özellikle yazılım geliştirme ile uğraşacak kişi ve kurumları destekleyecek politikalar uygulanmalıdır.
- Yazılım üretimini özendirecek tedbirler bir an önce yürürlüğe konulmalıdır.
- Yaygın olarak kullanılan genel amaçlı yazılım ürünleri ve diğer uluslararası yazılımlar Türkçe'ye çevrilmeli, Türkçe içerik geliştirilmelidir.
- TSE, uluslararası yazılım standartlarını, Türk standartlarına kazandırmalıdır.
- Yazılım pazarında Türkiye, Doğu-Batı arasında köprü görevini üstlenmelidir. Batıda geliştirilen yazılımlar için Avrasya pazarına destek hizmeti sunacak ülke olunması hedeflenmelidir. Bu hedefe yönelik yerelleştirme ve ara üretim mekanizmaları kurulmalıdır.
- Yazılım telif hakları ile ilgili mevzuat düzenlemeli ve yürürlüğe konmalıdır.
- Ülke çapında yazılım geliştirme/satın alma konularında danışma ve uzmanlık kurulları oluşturularak, standartlaşma ve teknolojik anlamda yönlendirme sağlanmalıdır.
- Üniversitelerde sertifika programları açılarak istihdam fazlası nitelikli işgücü yazılım sektörüne kazandırılmalıdır.
- Üniversiteleri daha fazla iş piyasası odaklı yapabilmek için gerekli düzenlemeler yapılmalı ve iş piyasası ile eğitim arasındaki bağlar geliştirmelidir.

5. UYGULAMA

Çalışmanın bu kısmında yazılım dili seçimi için ÇKKV yöntemleri olan AHP ve ANP sırasıyla uygulanmış ve iki yönteme göre elde edilen sonuçlar karşılaştırılmıştır.

5.1. Yazılım Dili Seçimi Karar Sürecine Etki Eden Kriterler ve Alternatifler

5.1.1. Kriterler

Karar sürecinde etkili olan kriterler literatür çalışmasından faydalanılarak tespit edilmiş, yine literatür çalışmasından faydalanılarak gruplandırılmıştır. Bu kriterler daha çok akademik kuruluşların, eğitim kurumlarının ve öğrencilerin yazılım dili seçiminde göz önüne tuttukları kriterlerdir. Belirlenen 20 adet kriter 8 grup altında toplanmış, kriterlerle ilgili açıklama yapılmıştır [32].

Akademik Kabulü Kriter Kümesi

Akademik Kabul: Akademik kabul kriteri, dilin Akademik kuruluşlarda tercih edilir olması ile ilgilidir. Nesne yönelimli yazılımın popülaritesi nedeniyle akademik kuruluşların çoğu nesneye yönelimli programlama dillerini kurslarına dahil etmişlerdir [32, 33].

Kitapların Bulunması: Kitapların bulunması faktörü diğer faktörden etkilenmektedir. Programlama dilinin yaşam döngüsündeki yeri kitap ve kaynak bulmayı oldukça etkilemektedir. Özellikle de yaşam döngüsünün başında olan yeni diller için bu faktör oldukça etkilidir. Yeni bir dil için kaliteli bir kitap bulmak çoğu zaman zordur. Dil geliştikçe kaynak bulmakta kolaylaşır. Akademik kabul kriterinin de kaynak kitapların bulunmasında önemli rolü vardır. Kullanılan öğrenme yaklaşımı da kaynak kitap bulmayı etkileyen kriterler arasındadır. Kaynak kitap bulma faktörü göz önüne alınması gereken önemli faktörler arasındadır[32, 34].

Öğrenci/ Akademik Versiyonu: Öğrenci veya Akademik versiyonunun olması öğrencilere bu programı kişisel bilgisayarlarına yükleme olanağı tanır. Böylelikle bilgisayar laboratuarları kapalı iken rahatlıkla çalışabile imkanına kavuşurlar. Eğer öğrenci versiyonuna ulaşmak mümkün değilse çalışma saatleri ve sayısı kısıtlı olan kampus bilgisayarlarından faydalanmakta zorlanacaklardır. Kampüslere ulaşımın da çok rahat olmadığını göz önüne alırsak bu faktörün öğrenciler için gözardı edilmemesi gerektiği aşikardır [32].

Endüstride Yaygınlığı Kriter Kümesi

Yaşam Döngüsündeki Safhası: Bir programlama dilinin yaşam döngüsündeki yeri ders kitapları ve kaynak bulmanın yanı sıra akademik kuruluşlarda ve endüstride yaygın olarak kullanılmasını da etkiler. Üniversiteler yaşam döngüsünün başlarında olan dilleri, yaşam döngüsünün sonunda popülaritesini kaybetmiş dillere tercih ederler.

Endüstride Kabulü: Endüstriyel kabul kriteri belli bir dilin piyasada yani endüstride ya da iş hayatında yaygınlığı ile ilgilidir. King (1992) birçok dil seçim kararının, o dilin popülaritesine veya gelecekte popüler olma ihtimaline bakılarak yapıldığını belirtmiştir [32,35]. Howland (1997) da birçok dilin pedagojik ve diğer nedenlerden çok, mevcut popülaritesi nedeniyle seçildiğini vurgulamıştır [32, 36].

Bilenlere Olan Talep: Dil seçimi genellikle iş yerinin ya da iş verenin isteğine göre yapılmaktadır. Öğrenciler tercih edilen bilgi ve becerilere sahip olma, iş bulma şanslarını arttıran bir dili öğrenme konusunda da daha isteklidirler [32,37]. Genellikle bir öğrencinin bir programı öğrenmeye başlamasıyla o programla ilgili çalışmaya başlaması arasında 4-5 yıl geçmektedir. Eğer bir müfredat öğrenciye çok yeni bir programlama dilini öğretirse öğrenci mezun olup iş hayatına atıldığında iş verenlerin hala bu dili bilenleri arayacağının bir garantisi yoktur. Programlama dilini seçerken endüstriyel kabul kriterini uzun vadede değerlendirmek kazanç sağlayacaktır.

Yazılım Özellikleri Kriter Kümesi

Sistem Gereksinimleri: Programlama dilinin sistem gereksinimleri seçim sürecinde rol oynar. Donanımın yanı sıra işletim sistemi gereksinimleri de vardır. Programlama dilini yüklemek için gereken hard disk alanı, işletim sistemi, programı çalıştırmak için gerekli bellek vb. dikkate alınması gereken faktörlerdir. Hem öğrencilerin hem de laboratuarların makineleri, seçilen dilin minimum gereksinimlerini karşılayabilmelidir [32].

İşletim Sistemi Bağımsızlığı: Bu kriter bir dilin belli bir işletim sistemi platformundan bağımsız olması ile ilgilidir. Örneğin .Net Framework'ü tarafından desteklenen diller Visual Basic, C++, C# vb. windows işletim sistemine dayanır. Java gibi diller ise platform bağımsızdır. Çeşitli işletim sistemlerinde kurulabilir. Platform bağımlı ya da bağımsız olmak iş verenin tercihine göre değişmektedir [32].

Açık Kaynak: Bu kriter uygulama geliştirme ortamının genişlemesini kontrol etmeyle ilgilidir. Örneğin Microsoft .Net framework tarafından desteklenen tüm dillerin ekleme, çıkarma, güncellemelerinden sorumludur. Sun Java dilinin gelişiminden sorumludur. Diğer taraftan PHP açık kaynak bir dil olarak açık kaynak topluluğuna üye herhangi biri tarafından kolaylıkla değiştirilebilir [32].

Kullanım Kolaylığı Kriter Kümesi

Uygulama Geliştirme Ortamını Kullanma Kolaylığı: Uygulama geliştirme ortamı programcının verimliliği geliştirebilen ya da engelleyebilen sanal çalışma tezgahıdır [32]. İyi tasarlanmış bir uygulama geliştirme ortamlarının öğrencilerin programı öğrenmesinde yardımcı olduğunu gösteren kanıtlar vardır [32,38]. Öğrencinin uygulama ortamından çok programlamaya konsantre olabilmesi için uygulama ortamının kolay kullanılabilir olması gerekmektedir [32,39].

Hata Bulma Kolaylığı: Bu kriter entegre uygulama geliştirme ortamının bir parçası olarak görülebilir. Bir programlama dilini değerlendirirken hata bulma kolaylıklarını

da göz önüne almak gerekir. Ad Hoc AP CS Komitesi (2000)'nin raporu, izleme ve hata bulma için programlama ortamının kapsamlı araçlar içermesi gerektiğini belirtmektedir. Hata tanıma net ve anlamlı olmalıdır [32,40].

Pedagojik Özellikleri Kriter Kümesi

Temel Kavramları Öğrenme Kolaylığı: Öğrenme eğrisi dilden dile ya da uygulama geliştirme ortamına göre büyük fark gösterir. Diller az ve öz sözdizimi ile kolayca anlaşılır şekilde nitelenmelidir [32,41].

Korumalı Kod Üretimi: Bu kriter iki önemli faktörü değerlendirmekte kullanılabilir. Birincisi dilin güvenilir tip kontrolü ve dizi sınırı kontrolü gibi özellikler sunup sunmaması, güvensiz değişken ve göstergelerden kaçınması ile ilgilidir. Kölling et al.(1995) bir dilde güvenli, sağlam tip sistemi olması, algılanamaz, ilk değeri atanmamış değişkenlerin olmaması gerektiğini belirtmiştir [32,39]. İkinci faktör ilki ile ilişkilidir, bir Java programının girebileceği adresleri kısıtlayan sandbox gibi güvenlikle ilgili özelliklerin dahil olmasıdır.

İleri Özellikler: Bir çok program temel programlama dili özelliklerini bir başlangıç müfredatında verir ve ileri özelliklerini ise sonraya bırakır. Bir programlama dilinin ileri özellikleri yeteri kadar destekleyip desteklemediği dikkate alınması gereken kriterler arasındadır [32].

Dilin Amacı Kriter Kümesi

Web Tabanlı Uygulama Geliştirme: Bu kriter bir dilin sağladığı web uygulaması geliştirme desteği ile ilgilidir. Bu durum ASP.Net gibi web uygulaması geliştirmede yüksek oranda destek sağlayan web uygulama geliştirme teknolojilerini de içermektedir. Günümüzde birçok program web-tabanlı uygulama geliştirme becerilerini öğrencilere kazandırmanın öneminin bilincindedir [32].

Spesifik Uygulama Destekleme: Bu kriter, özel uygulamalar için bir dilin programlamayı ne kadar iyi desteklediği ile ilgilidir[32,42]. Uygulama alanlarına örnek olarak, bilimsel amaçlı programlama için daha çok FORTRAN desteğini, iş veri süreçleri için COBOL'u, rapor oluşturmada RPG desteğini verebiliriz.

Dil Metodolojisi Kriter Kümesi

İfadeleri Anlama ve Yorumlama Yöntemi: Bu kriter bir programlama dilinin Soyutlama, Çok Biçimlilik(Polymorphism), Miras(Inheritance), Kapsülleme (Encapsulation) gibi temel nesneye yönelik kavramları ne kadar iyi değerlendirdiği ile ilgilidir. Eğitmenler bazı nesne yönelimli olduğunu belirten dillerin az önce bahsedilen özellikleri sağlamada yetersiz kaldığını göz önünde almalıdırlar [32].

Öğrenme Yaklaşımı: Bu kriter bir dilin tercih edilen öğrenme yaklaşımını ne kadar iyi desteklediğinin değerlendirilmesi ile ilgilidir. Öncelikle amacımızın dili bir araç olarak kullanarak programlama kavramını öğretmeyi mi desteklemek, yoksa belli bir dilin özelliklerini mi öğretmek olduğuna karar verilmelidir [32].

Bir programlama dilinin tümünü birden öğrenip sonra da kullanmaya çalışmak profesyonel programcılar için bile çok zordur. Programlama dilleri, getirdikleri olanaklar küçük örneklerle denenerek parça parça öğrenilir. Bir dili öğrenme şekli, programlamaya yeni başlayanlara etkin programlama tekniklerini de öğretecek şekilde olmalıdır. Öğretmenin en iyi yolu, iyi seçilmiş somut örneklerden başlayıp daha genel ve daha soyut örneklere geçmektir. Dilin olanakları her zaman için kullanıldıkları kapsamda sunulmalıdır. Yoksa programcının ilgisi, sistem üretmek yerine anlaşılması güç teknik ayrıntılara yönelir [43].

Dil Desteği ve Eğitim Kriter Kümesi

Desteğe Ulaşabilme: Bu kriter her şekilde destek olabilecek kişileri, araçları bulmakla ilgilidir. Satıcı desteğine ek olarak, internetten ve forumlardan sağlanacak

desteğin de göz önüne alınması gerekmektedir [32,44]. Eğitici rehberleri, örnek programlar, öğrenciler için alıştırma kitapları vb.

Eğitmenlerin/Personelin Eğitimi: Bu eğitmenler ve destek personeli için gerekli eğitim, bir dili veya uygulama geliştirme ortamını öğrenmek için gereken zaman, nitelikli eğitmenlerin bulunabilmesi ile ilgili kriterdir. Teknolojik ortamın gerektirdiği niteliklere sahip insan gücünü oluşturma oldukça önem taşımaktadır. İyi eğitilmiş insan gücünün değeri ve gerekliliği aşikardır. Yeni bir dili benimsemek gönüllülük ister çünkü eğitmenler hem kendileri hem de öğrencileri için yeni olan bir dili öğretirken sürekli kendi niteliklerini zenginleştirmeli, yeni öğrenme metotlarının ve pratik metotların eklenmiş olduğu teknikleri uygulamalıdırlar [32,45].

5.1.2. Alternatifler

Nesne yönelimli programlama tekniği, diğer yaklaşımlara nazaran, yazılım geliştiren insanlara büyük avantajlar sağlamış ve günümüzde yaygın olarak kullanılmaya başlamıştır. Bu çalışmada uzman görüşlerine danışılarak günümüzde en çok tercih edilen nesne yönelimli dillerden C++, C# , Delphi ve Java dilleri alternatif olarak ele alınmıştır.

C++ Programlama Dili

1980'lerin başında yine Bell Laboratuarlarında bir araştırmacı olan Bjarne Stroustrup doktora çalışmalarında çalıştığı bazı "nesne tabanlı" araçların özelliklerini, özellikle sınıfları destekleyen bir C sürümü geliştirmeye çalışıyordu. C dilinin üreticileri olan Brian Kernighan ve Dennis Ritchie'nin de dahil olduğu bir grup tarafından desteklenmekteydi. Bu yeni sürüme cfront adını vermişti. Çalışmasının potansiyeli kısa sürede keşfedildi ve 1983'de C++ ortaya çıktı. 1990'ların başına kadar, C'nin gelişme sürecine çok benzer bir süreç ile geliştirildi. 1990'ların başında C++'ın geliştirilmesi önce ANSI daha sonra da ISO gibi standart geliştirme kuruluşlarına devredildi [46].

C++ tamamen nesne yönelimli bir programlama dili değildir, nesne yönelimli programlamayı destekler ve nesne yönelimli olmayan kısımları da vardır.

C++ öğrenmekle ilgili olarak anlatılan kötü anıların büyük çoğunluğu C++'ı nasıl öğrenmesi gerektiğini bilmeden öğrenmeye çalışan insanların düştükleri hatalardan kaynaklanır. C++ nesne tabanlı programlama da dahil olmak üzere tam dört değişik paradigma ile program yazmaya olanak veren özel bir dildir. Bu paradigmalar istenildiği gibi karışık da kullanılabilir. İşte bu çok paradigmalı yaklaşım C++ ile olağanüstü çözümlerin hem de şık bir kodlama stili ile geliştirilmesine olanak verir. Ancak programlamaya yeni başlayan ve bu paradigmaların farklarını bilmeyen bir kişi için bu karışık stil anlaşılması çok zor bir şeydir. C++ öğrenmek isteyen kişinin, en azından bir süre için, tek bir paradigmada sabit kalması gerekir. Daha sonra diğer paradigmaları nasıl kullanacağını görmelidir. En sonunda da paradigmaları karışık ve birlikte kullanmaya başlayacaktır [46].

C# Programlama Dili

Borland firmasından ayrılan, programlama efsanesi Anders Hejlsberg, 1996'da Microsoft'a katılarak 2000'li yılların başında C# dilini geliştirmiştir.

Web teknolojisi gözönünde tutularak .NET platformu için tasarlanmış bir dildir. C# ile Web tabanlı uygulamaların dışında farklı uygulamalar da gerçekleştirmek mümkündür. C# programcıya bazı kolaylıklar getirmiş, nesneye dayalı yeni bir dildir. İleriye dönük olarak Microsoft'un diğer dillerin iyi özellikleri alıp harmanlayarak ortaya sürdüğü bir programlama dilidir . C# dili C++ ve JAVA'ya benzemektedir. C#'ın mimarı Anders Hejlsberg C#'ı tasarlarken özellikle C++'a yakın durmaya çalıştıklarını belirtmiştir [47].

C# öğrenilmesi kolay bir dildir. C# nesne yönelimli programlama diline tam destek verir. Kodlama hatalarını azaltıcı önlemleri göz önüne alarak programcıların verimliliğinin artmasını sağlamıştır. C#'ta herşey bir sınıftır, değişken kavramı neredeyse kalkmıştır. Bu şu anlama gelmektedir: Yazılan her kod parçası bir sınıf

dosyasının içinde bulunur. C# dili güç ve hızlılık arasındaki dengeyi estetik bir şekilde korumaktadır. NET sınıf kütüphanesinin büyük bir kısmı C# ile geliştirilmiştir. Bu kütüphane en etkin biçimde C# ile kullanabilir. C# dili .NET'in çalışma mimarisi de gözönünde bulundurularak sıfırdan tasarlandığı için .NET'in bütün olanaklarından en etkin biçimde faydalanır. C#, modern programlama tekniklerine tam destek veren, hızlı ve etkin bir şekilde kodlama yapılabilen bir programlama dilidir.

Java Programlama Dili

Java ilk olarak 1991 yılında Sun Microsystems'te görevli James Gosling and Patrick Naughton tarafından Green kod adıyla, TV ve mikrodalga fırın gibi elektronik ev eşyalarını kontrol etmekte ve bilgi alış verişinde kullanılacak bir programlama dili olarak tasarlandı. Ancak bu teknoloji bir tek müşteri bile elde edemedi. Çok geçmeden Sun elektronik ev eşyalarındaki pazarın pek de istekleri doğrultusunda gelişmeyeceğini anladı. O sıralar bilişim dünyasında adı sık sık telafuz edilir olan Internette projesinin geleceği için potansiyel gören Sun, önce programlama dilini Oak olarak yeniden isimlendirdi ve projesini bu yönde geliştirme kararı aldı. Daha sonra Hotjava isimli web browser Oak kullanılarak geliştirildi. Sonra Oak teknolojisi Java olarak yeniden isimlendirildi ve Java'nın, entegre sistem çatıları oluşturmada en çok tercih edilen teknoloji ve programlama dillerinden biri olduğu bugünlere gelindi. Java geliştirilmesindeki ilk hedef olan ev eşyaları alanına Java destekli cep telefonlarıyla dönüş yapmıştır. Java çeşitli alanlarda uygulamalar geliştirebilen dinamik bir dildir. Ayrıca platform bağımsızdır. Java ile yazılan kodlar her makinede aynı şekilde çalışır [48].

Java tam anlamıyla nesne yönelimli bir dil olarak geliştirilmiştir. Nesne yönelimli programları oluşturmak, yönetmek ve kontrol altında tutmak daha kolaydır. Java bu mantığın dışına çıkacak özellikte kötü programlama alışkanlıklarına izin vermemektedir.

Delphi Programlama Dili

Pascal tabanlı bir dil olup nesne yönelimli programlama yapabilme özelliği taşır. Öğreniminin çok zor olmayışı ve üniversitelerde pascal eğitiminin ağırlıklı verilmesi nedenleriyle çoğu bilgisayar programlama öğrencisinin tercih ettiği bir dildir.Visual programlama özelliği taşır.

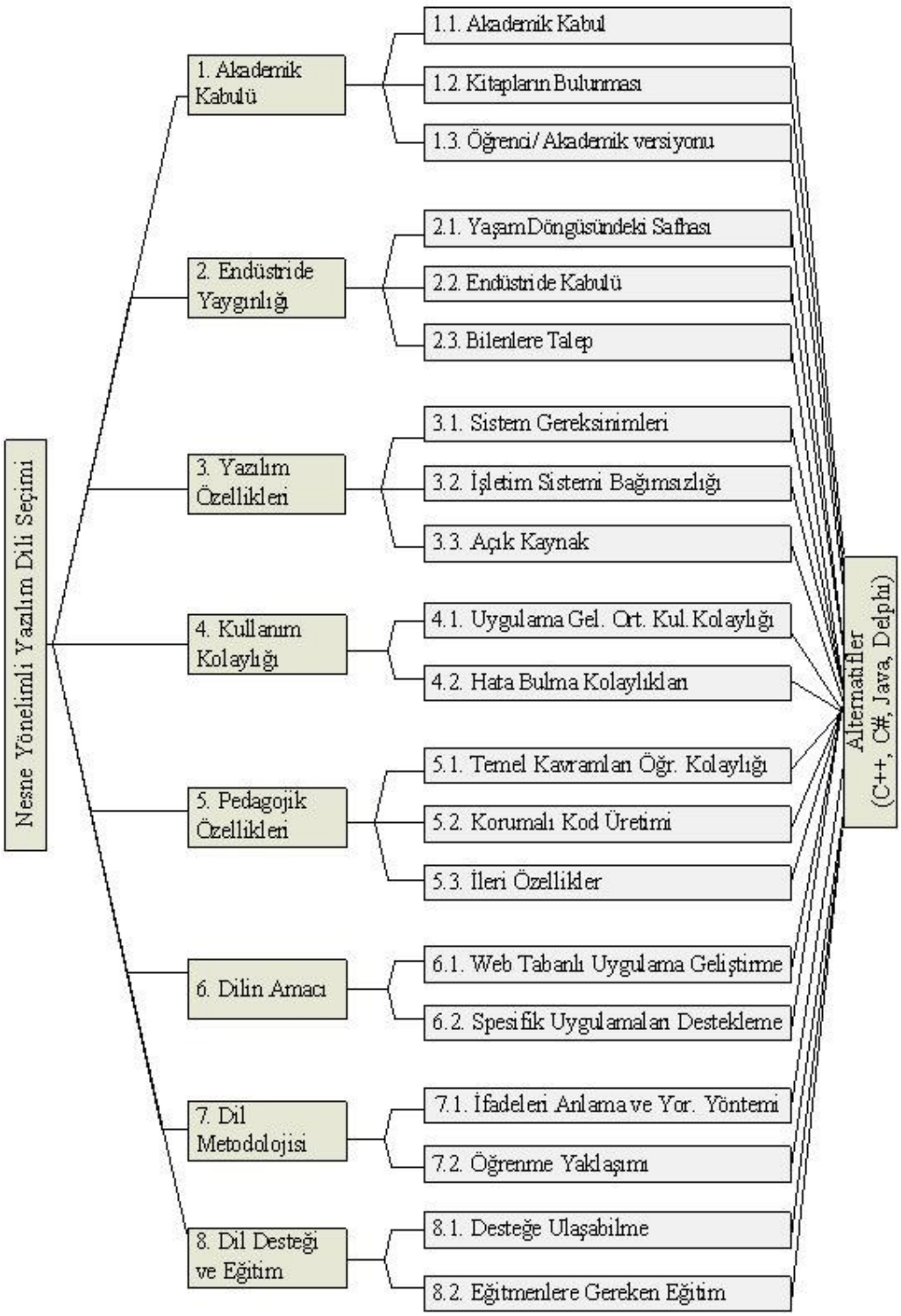
Windows platformu için masaüstü uygulamaları geliştiren programcıların, geçmişte bugün olduğu kadar çok alternatifi yoktu. Delphi, Visual C++'a yaklaşan gücü ve neredeyse Visual Basic kadar kolay olmasıyla programcıların gözdesi haline geldi. Delphi'de hiçbir geliştirme aracında olmayan bir başka özellik daha vardı: Visual Component Library (VCL) adı verilen devasa bir bileşen kütüphanesi. Grafik çizmek, e-posta göndermek, ağ üzerindeki makinalara erişmek, Explorer ağacı görüntüsü vermek, ses dosyalarını çalmak, veritabanında yatan bilgilere erişmek, internet protokollerini kullanmak, değişik dosya formatlarını okuyabilmek ve çok daha fazlası için hazır bileşenler Delphi ile birlikte geliyordu.

Delphi, bu özelliğini günümüzde de genişleterek sürdürmektedir. Ayrıca Linux üzerinde çalışan kardeşi Kylix sayesinde, belli standartlara sadık kalan Delphi uygulamalarını Linux üzerinde çalıştırmak da mümkündür. Bunun yanı sıra Delphi de, .NET platformu üzerinde uygulama geliştirilebilecek dillerden biri haline gelmiştir. Uygulamalarını .NET, JVM gibi bir platformun yüklü olmasına gerek duymadan her Windows makinasında çalıştırmak isteyen ve hem kolay, hem de güçlü bir geliştirme ortamı arayan programcıların tercihidir [49].

5.2. AHP ile Yazılım Dili Seçimi

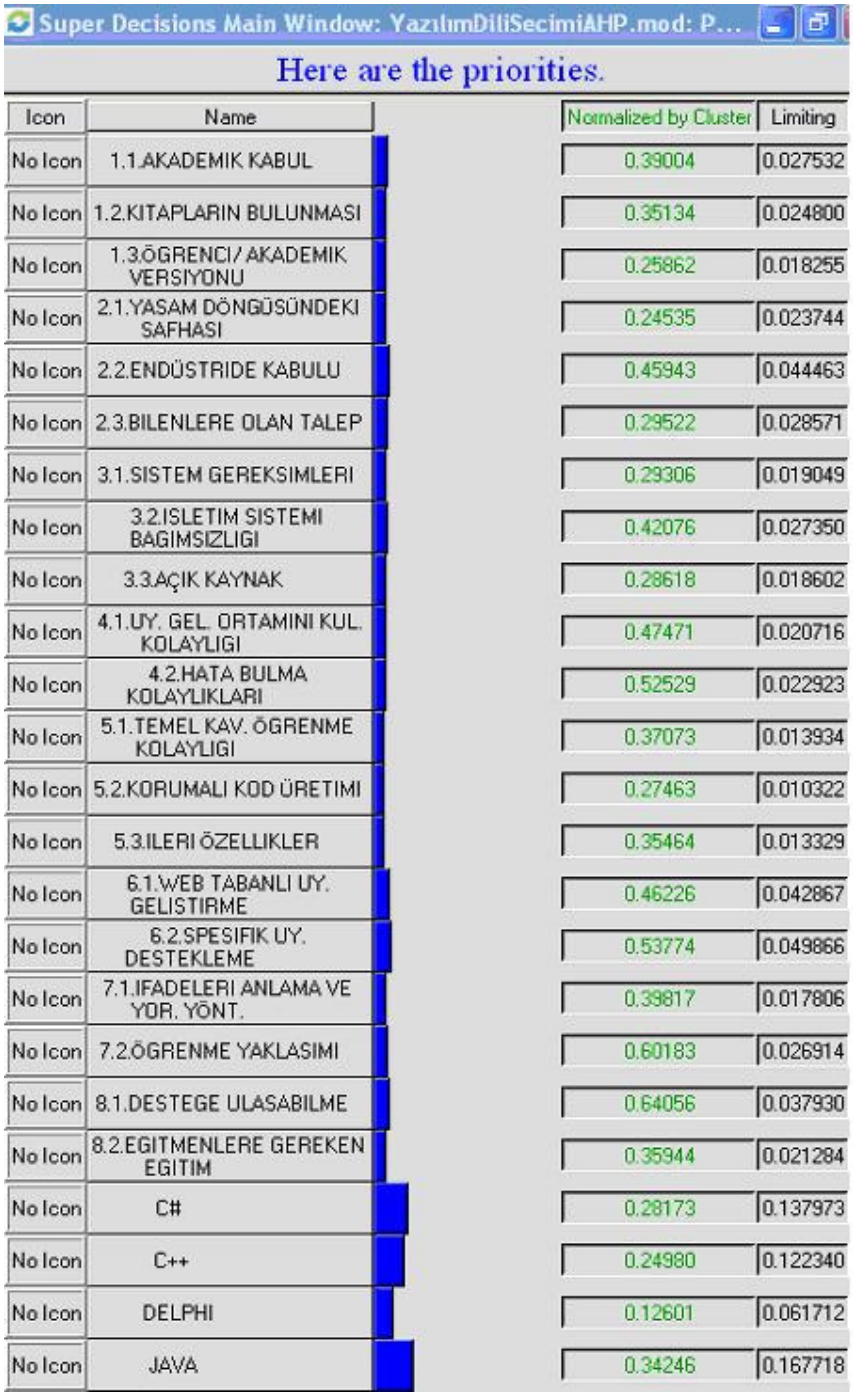
AHP, karar vericinin belirlediği her bir kriterin göreceli önemlerini belirler ve daha sonra her bir kritere göre karar alternatifleri arasında seçim yapar. Hiyerarşik yapıya sahip AHP'de kriterler arasında bağımlılık söz konusu değildir. AHP prensipleri doğrultusunda öncelikle amaç, bunu takip eden ölçütler ve alternatifler gösterilmiştir. AHP'nin öngördüğü şekilde problemin şematik olarak yapılandırılmış hali Şekil

5.1'de görülmektedir. Modelimizde sekiz adet kriter kümesi, toplamda yirmi adet kriter dikkate alınarak hem dört alternatif arasında değerlendirme yapılacak hem de kriterler arasında önem derecelerine göre bir sıralama yapılacaktır.



Şekil 5.1. Yazılım dili seçimi için AHP modelinin şematik gösterimi

AHP ile kurulmuş olan model ikili karşılaştırma anketlerinden faydalanılarak Super Decisions 1.6.0 programı ile çözümlenmiş olup elde edilen alternatif ve kriter ağırlıkları Şekil 5.2'de görülmektedir.



Icon	Name	Normalized by Cluster	Limiting
No Icon	1.1.AKADEMIK KABUL	0.39004	0.027532
No Icon	1.2.KITAPLARIN BULUNMASI	0.35134	0.024800
No Icon	1.3.ÖGRENCI/ AKADEMIK VERSIYONU	0.25862	0.018255
No Icon	2.1.YASAM DÖNGÜSÜNDEKI SAFHASI	0.24535	0.023744
No Icon	2.2.ENDÜSTRIDE KABULU	0.45943	0.044463
No Icon	2.3.BILENLERE OLAN TALEP	0.29522	0.028571
No Icon	3.1.SISTEM GEREKSIMLERI	0.29306	0.019049
No Icon	3.2.ISLETIM SISTEMI BAGIMSIZLIGI	0.42076	0.027350
No Icon	3.3.AÇIK KAYNAK	0.28618	0.018602
No Icon	4.1.UY. GEL. ORTAMINI KUL. KOLAYLIGI	0.47471	0.020716
No Icon	4.2.HATA BULMA KOLAYLIKLARI	0.52529	0.022923
No Icon	5.1.TEMEL KAV. ÖGRENME KOLAYLIGI	0.37073	0.013934
No Icon	5.2.KORUMALI KOD ÜRETIMI	0.27463	0.010322
No Icon	5.3.ILERI ÖZELLIKLER	0.35464	0.013329
No Icon	6.1.WEB TABANLI UY. GELISTIRME	0.46226	0.042867
No Icon	6.2.SPESIFIK UY. DESTEKLEME	0.53774	0.049866
No Icon	7.1.IFADELERI ANLAMA VE YÖR. YÖNT.	0.39817	0.017806
No Icon	7.2.ÖGRENME YAKLASIMI	0.60183	0.026914
No Icon	8.1.DESTEGE ULASABILME	0.64056	0.037930
No Icon	8.2.EGITMENLERE GEREKEN EGITIM	0.35944	0.021284
No Icon	C#	0.28173	0.137973
No Icon	C++	0.24980	0.122340
No Icon	DELPHI	0.12601	0.061712
No Icon	JAVA	0.34246	0.167718

Şekil 5.2. AHP çözümüne göre kriterlerin ve alternatiflerin limit öncelik değerleri

Şekil 5.2'de limit önceliklerinden (limiting değerleri) faydalanarak kriterler arasındaki öncelik sıralamasını elde edebiliriz. AHP çözümüne göre 0,0499 değeri ile *spesifik uygulamaları destekleme* kriteri birinci sırada yer alırken, ikinci sırada 0,0445 değeri ile *endüstride kabulü* kriteri yer almaktadir.Hemen ardından 0,0429 değeri ile *web tabanlı uygulama geliştirme kriteri* gelmektedir. Çizelge 5.1'de AHP çözümüne göre kriterler arasında elde edilen sıralama görülmektedir.

Çizelge 5.1. AHP çözümüne göre kriterler arasında elde edilen sıralama

Sıralama	**Kriterler**
1	6.2. Spesifik Uygulamaları Destekleme
2	2.2. Endüstride Kabulü
3	6.1. Web Tabanlı Uygulama Geliştirme
4	8.1. Desteğe Ulaşabilme
5	2.3. Bilenlere Talep
6	1.1. Akademik Kabul
7	3.2. İşletim Sistemi Bağımsızlığı
8	7.2. Öğrenme Yaklaşımı
9	1.2. Kitapların Bulunması
10	2.1. Yaşam Döngüsündeki Safhası
11	4.2. Hata Bulma Kolaylıkları
12	8.2. Eğitmenlere Gereken Eğitim
13	4.1. Uygulama Gel. Ortamını Kul.Kolaylığı
14	3.1. Sistem Gereksinimleri
15	3.3. Açık Kaynak
16	1.3.Öğrenci/ Akademik Versiyonu
17	7.1. İfadeleri Anlama ve Yor. Yöntemi
18	5.1. Temel Kavramları Öğr. Kolaylığı
19	5.3. İleri Özellikler
20	5.2. Korumalı Kod Üretimi

Şekil 5.2'den tüm alternatiflerin limit önceliklerinin normalize olmuş değerlerine(Normalized by Cluster) bakılarak elde edilen sıralama Çizelge 5.2'deki gibidir.

Çizelge 5.2. AHP çözümüne göre yazılım dilleri arasındaki sıralama

1	Java	%34,24
2	C#	%28,17
3	C++	%24,98
4	Delphi	%12,60

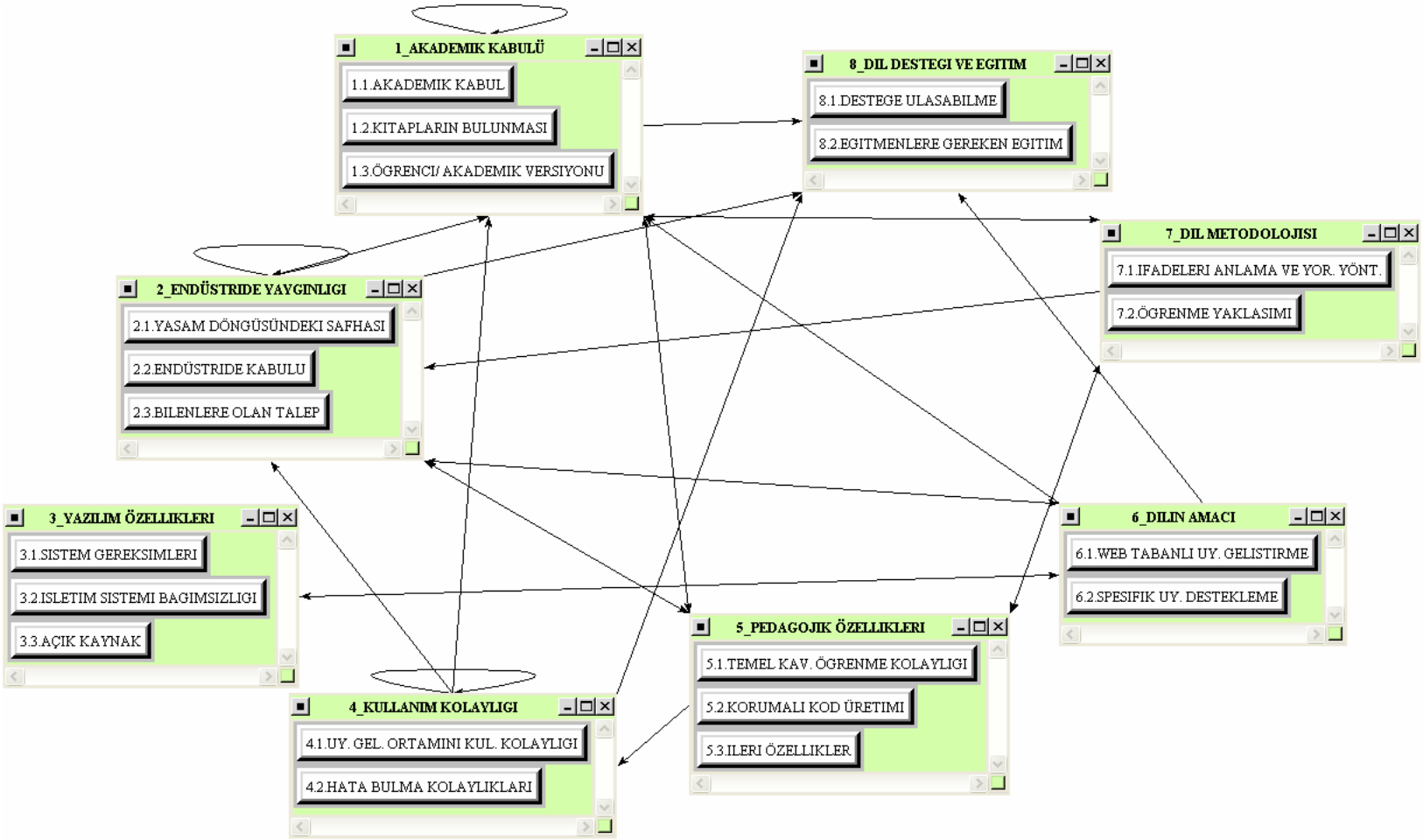
Belirlemiş olduğumuz kriterlere göre alternatiflerin aldıkları değerleri incelediğimizde ilk sırada Java görülmektedir. İkinci sırada ise C# yazılım dili vardır. Hemen peşinden C# dili ve son sırada ise Delphi dili gelmektedir.

5.3. ANP ile Yazılım Dili Seçimi

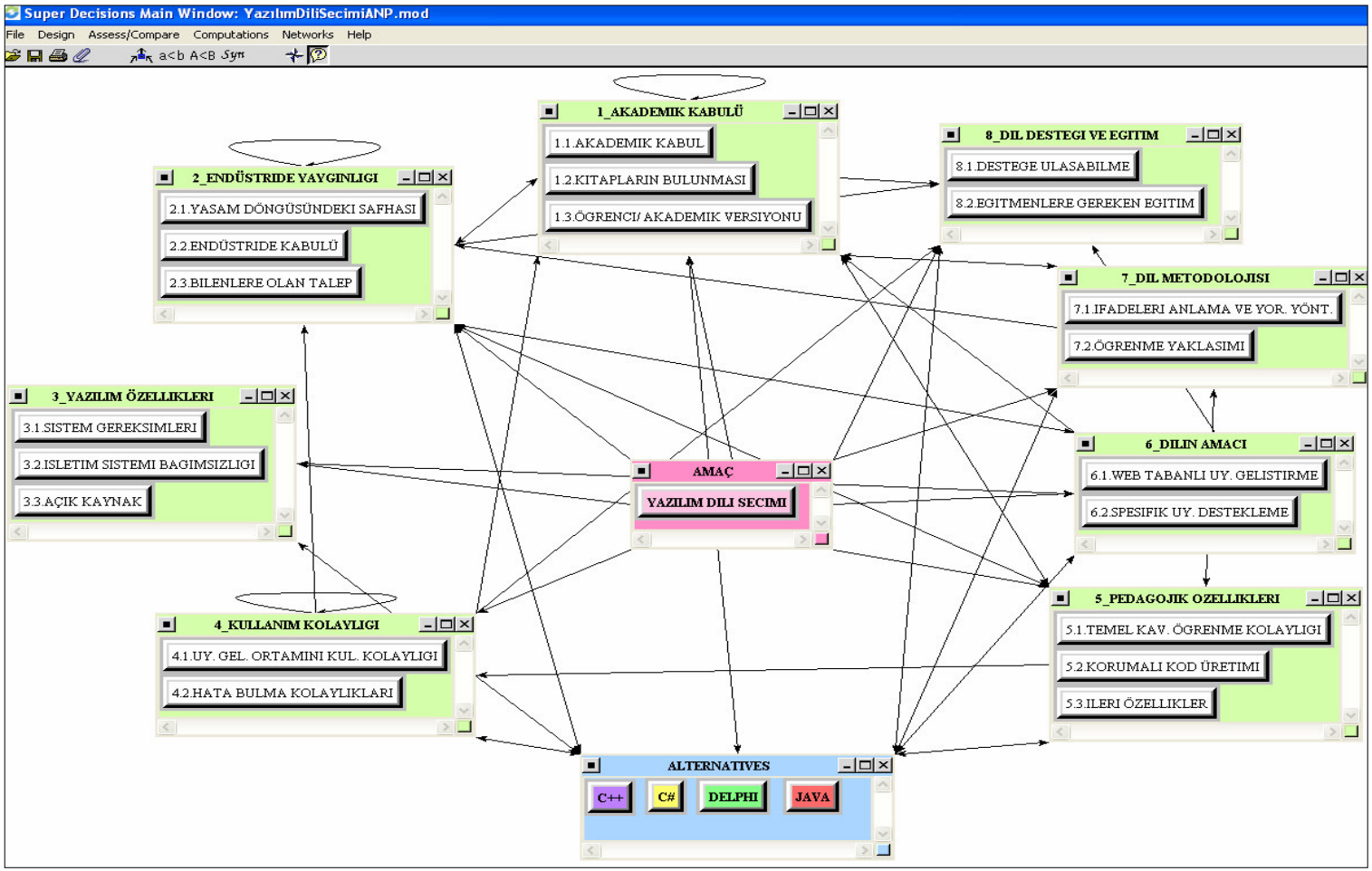
Saaty kriterler arasında bağımlılığın söz konusu olduğu durumlarda daha tutarlı bir sonuç elde edilmesi için ANP tekniğinin kullanılmasını önermiştir.

5.3.1. Ağ yapısı ve ilişkilerin tanımlanması

ANP'de kullanılan kriter kümeleri ve kriterler arasındaki ilişkileri göstermek için öncelikle network yapısı oluşturulur. Modelimizde uzman görüşlerine göre Super Decisions programında oluşturduğumuz network yapısı Şekil 5.3'te verilmiştir. Bu network yapısı kriter kümeleri ve kriterler arasındaki iç ve dış bağımlılıklara göre oluşturulmuştur. Okların yönleri bağımlılığı ortaya koyar. Ok yönü etkileyen bileşenden etkilenen bileşene doğrudur. Bir kriter kümesinin kendi içindeki kriterler arasında bağımlılık yani iç bağımlılık söz konusuysa bu durum kriter kümesinde çıkan okun yine aynı kriter kümesine dönmesi şeklinde gösterilir. Modelimizde *akademik kabul*, *endüstride yaygınlığı* ve *kullanım kolaylığı* kriter kümelerinde iç bağımlılık söz konusudur. ANP'nin öngördüğü şekilde problemin tam olarak yapılandırılmış hali Şekil 5.4'te görülmektedir.



Şekil 5.3. Kriter kümeleri ve kriterler arasındaki ilişki



Şekil 5.4.Yazılım dili seçimi için ANP modeli

Aşağıdaki çizelgede kriterlerin birbirlerini etkileme durumu belirtilmiştir.

Çizelge 5.3. Kriterler arasındaki etkileşim

Etkileyen Kriterler	Etkilenen Kriterler
1.1. Akademik Kabul	1.2. Kitapların Bulunması 1.3.Öğrenci/ Akademik versiyonu
	2.1. Yaşam Döngüsündeki Safhası 2.2. Endüstride Kabulü 2.3. Bilenlere Talep
	8.1. Desteğe Ulaşabilme 8.2. Eğitmenlere Gereken Eğitim
1.2. Kitapların Bulunması	1.1. Akademik Kabul
	5.1. Temel Kavramları Öğr. Kolaylığı
	7.1. İfadeleri Anlama ve Yor. Yöntemi 7.2. Öğrenme Yaklaşımı
	8.1. Desteğe Ulaşabilme 8.2. Eğitmenlere Gereken Eğitim
1.3.Öğrenci/ Akademik versiyonu	1.1. Akademik Kabul 1.2. Kitapların Bulunması
	5.1. Temel Kavramları Öğr. Kolaylığı
	8.1. Desteğe Ulaşabilme 8.2. Eğitmenlere Gereken Eğitim
2.1. Yaşam Döngüsündeki Safhası	1.1. Akademik Kabul 1.2. Kitapların Bulunması 1.3.Öğrenci/ Akademik Versiyonu
	2.2. Endüstride Kabulü 2.3. Bilenlere Talep
	5.3. İleri Özellikler
	8.1. Desteğe Ulaşabilme 8.2. Eğitmenlere Gereken Eğitim

Çizelge 5.3. (Devam) Kriterler arasındaki etkileşim

2.2. Endüstride Kabulü	1.3. Kitapların Bulunması
	2.1. Yaşam Döngüsündeki Safhası
	2.3. Bilenlere Talep
	6.2. Spesifik Uygulamaları Destekleme
	8.1. Desteğe Ulaşabilme
3.2. İşletim Sistemi Bağımsızlığı	6.2. Spesifik Uygulamaları Destekleme
4.1. Uygulama Geliştirme Ortamını Kullanma Kolaylığı	1.1. Akademik Kabul
	1.2. Kitapların Bulunması
	2.2. Endüstride Kabulü
	4.2. Hata Bulma Kolaylıkları
	8.1. Desteğe Ulaşabilme
	8.2. Eğitmenlere Gereken Eğitim
4.2. Hata Bulma Kolaylıkları	4.1. Uygulama Geliştirme Ortamını Kullanma Kolaylığı
5.1. Temel Kavramları Öğr. Kolaylığı	1.1. Akademik Kabul
	1.2. Kitapların Bulunması
	7.1. İfadeleri Anlama & Yor. Yönt.
	7.2. Öğrenme Yaklaşımı
5.2. Korumalı Kod Üretimi	1.1. Akademik Kabul
	2.2. Endüstride Kabulü
5.3. İleri Özellikler	4.1. Uygulama Gel. Ortamı Kul.Kolaylığı
	4.2. Hata Bulma Kolaylıkları
6.1. Web Tabanlı Uygulama Geliştirme	1.2. Kitapların Bulunması
	2.2. Endüstride Kabulü
	2.3. Bilenlere Talep
	3.1. Sistem Gereksinimleri
	8.1. Desteğe Ulaşabilme
7.1. İfadeleri Anlama & Yor. Yöntemi	1.1. Akademik Kabul
	1.2. Kitapların Bulunması

Çizelge 5.3. (Devam) Kriterler arasındaki etkileşim

7.2. Öğrenme Yaklaşımı	1.1. Akademik Kabul 1.2. Kitapların Bulunması
	2.1. Yaşam Döngüsündeki Safhası
	5.1. Temel Kavramları Öğr. Kolaylığı

5.3.2. İkili karşılaştırmaların yapılması

İkili karşılaştırmaların elde edilebilmesi için üç farklı üniversiteden birer akademisyenin ve Havelsan'dan iki bilgisayar mühendisinin görüşleri alınarak beş uzman kişiye anket formları ayrı ayrı doldurulmuştur. Çalışmanın daha objektif olması açısından her kurumdan konusunda uzman bir ya da iki kişinin katılımı yeterli görülmüştür. Sonuçların daha tutarlı olması açısından kişilere anketler bilgisayar başında yapılmış, her ekranın tutarlılığı kontrol edilmiş ve tutarlılık oranı tüm karşılaştırmalar için sağlanmıştır. Çalışma ile ilgili kullanılan anketler EK-1'de verilmiştir.

Beş kişiden gelen anket sonuçlarının geometrik ortalaması alınarak her bir ikili karşılaştırma matrisi için hesaplanan değerler Super Decisions 1.6.0 programına manuel olarak girilmiştir.

İkili karşılaştırmalar üç temel aşamada yapılmıştır. Birinci aşamada kriter kümeleri karşılaştırılmış, ikinci aşamada her bir kriter tek tek ele alınıp iç ve dış bağımlılıklarına göre ilgili kriterler arasında karşılaştırılmalar yapılmış ve tüm alternatifler bu kriter bazında karşılaştırılmış, üçüncü yani son aşamada ise her bir alternatif için tüm kriterler karşılaştırılmıştır.

5.3.3. Matrislerin elde edilmesi

Tüm karşılaştırma matrislerine değerler girildikten sonra Super Decisions programı ile hesaplanan ağırlıklandırılmamış matris (Şekil 5.5), ağırlıklandırılmış matris (Şekil 5.6) ve limit matris (Şekil 5.7) elde edilmiştir.

Ağırlıklandırılmamış matris, ağ üzerinde tüm elemanların arasındaki direkt ya da dolaylı etkileşimi yapılan ikili karşılaştırmaları gösterir.

Ağırlıklandırılmış matris, ağırlıklandırılmamış matrisin stokastik yani sütun toplamlarının bire eşit hale dönüştürülmüş biçimidir.

Limit matris, ağırlıklandırılmış matrisin satırları durağan hale gelinceye kadar kuvvetinin alınmasıyla elde edilir. Limit matris uygun alternatifi belirtmekle kalmaz, her bir kriterin karar sürecindeki önemini ve katkısını da gösterir.

Super Decisions Main Window: YazılımDiliSecimi.mod: Unweighted Super Matrix

	1.1.AKA~	1.2.KIT~	1.3.ÖGR~	2.1.YAS~	2.2.END~	2.3.BIL~	3.1.SIS~	3.2.ISL~	3.3.AÇI~	4.1.UY.~	4.2.HAT~	5.1.TEM~	5.2.KOR~
1.1.AKA~	0.00000	1.00000	0.44249	0.40998	0.00000	0.00000	0.00000	0.00000	0.00000	0.45454	0.00000	0.16023	1.00000
1.2.KIT~	0.76191	0.00000	0.55751	0.33069	1.00000	0.00000	0.00000	0.00000	0.00000	0.54546	0.00000	0.83977	0.00000
1.3.ÖGR~	0.23810	0.00000	0.00000	0.25933	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2.1.YAS~	0.11500	0.00000	0.00000	0.00000	0.31248	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2.2.END~	0.70492	0.00000	0.00000	0.50000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	1.00000
2.3.BIL~	0.18008	0.00000	0.00000	0.50000	0.68752	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3.1.SIS~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3.2.ISL~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3.3.AÇI~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
4.1.UY.~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.35463	0.00000	0.00000
4.2.HAT~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	1.00000	0.64537	0.00000	0.00000
5.1.TEM~	0.00000	1.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
5.2.KOR~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
5.3.ILE~	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
6.1.WEB~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
6.2.SPE~	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000
7.1.IFA~	0.00000	0.52381	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.80769	0.00000
7.2.ÖGR~	0.00000	0.47619	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.19231	0.00000
8.1.DES~	0.59016	0.47619	0.13330	0.82517	1.00000	0.00000	0.00000	0.00000	0.00000	0.54546	0.00000	0.00000	0.00000
8.2.EGI~	0.40984	0.52381	0.86670	0.17483	0.00000	0.00000	0.00000	0.00000	0.00000	0.45454	0.00000	0.00000	0.00000
C#	0.19386	0.24675	0.17241	0.35719	0.26056	0.31073	0.31209	0.13497	0.22403	0.37335	0.39684	0.42197	0.41292
C++	0.29914	0.27571	0.35520	0.22248	0.25058	0.21968	0.21814	0.27339	0.17233	0.08853	0.14098	0.27587	0.16721
DELPHI	0.06580	0.08203	0.08552	0.07004	0.09006	0.08000	0.13308	0.06900	0.08528	0.41325	0.30876	0.09035	0.15079
JAVA	0.44120	0.39551	0.38688	0.35029	0.39879	0.38959	0.33669	0.52264	0.51836	0.12488	0.15341	0.21181	0.26908
YAZILIM~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

Şekil 5.5. Ağırlıklandırılmamış matris

	5.3.ILE~	6.1.WEB~	6.2.SPE~	7.1.IFA~	7.2.ÖGR~	8.1.DES~	8.2.EGI~	C#	C++	DELPHI	JAVA	AMAÇ
1.1.AKA~	0.00000	0.00000	0.00000	0.52381	0.66667	0.00000	0.00000	0.31993	0.40582	0.28940	0.47326	0.32609
1.2.KIT~	0.00000	1.00000	0.00000	0.47619	0.33333	0.00000	0.00000	0.45520	0.34697	0.52830	0.20396	0.29350
1.3.ÖGR~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.22487	0.24721	0.18231	0.32278	0.38041
2.1.YAS~	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.28659	0.32814	0.17351	0.17746	0.38758
2.2.END~	0.00000	0.54546	0.00000	0.00000	0.00000	0.00000	0.00000	0.51344	0.47081	0.55806	0.37040	0.40035
2.3.BIL~	0.00000	0.45454	0.00000	0.00000	0.00000	0.00000	0.00000	0.19997	0.20105	0.26843	0.45214	0.21207
3.1.SIS~	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.44523	0.29036	0.38924	0.11770	0.43586
3.2.ISL~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.35577	0.47094	0.28482	0.49230	0.26193
3.3.AÇI~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.19900	0.23870	0.32594	0.39000	0.30221
4.1.UY.~	0.50000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.40001	0.33333	0.69697	0.55752	0.29413
4.2.HAT~	0.50000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.59999	0.66667	0.30303	0.44248	0.70587
5.1.TEM~	0.00000	0.00000	0.00000	0.00000	1.00000	0.00000	0.00000	0.42188	0.26316	0.37883	0.40322	0.41226
5.2.KOR~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.42188	0.26317	0.35802	0.13377	0.31291
5.3.ILE~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.15625	0.47367	0.26316	0.46301	0.27482
6.1.WEB~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.71429	0.14288	0.71591	0.80000	0.47619
6.2.SPE~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.28571	0.85712	0.28409	0.20000	0.52381
7.1.IFA~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.33333	0.33333	0.66667	0.40001	0.61538
7.2.ÖGR~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.66667	0.66667	0.33333	0.59999	0.38461
8.1.DES~	0.00000	1.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.33333	0.71429	0.75000	0.85714	0.75000
8.2.EGI~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.66667	0.28571	0.25000	0.14286	0.25000
C#	0.26020	0.40120	0.15409	0.38107	0.33770	0.25390	0.11156	0.00000	0.00000	0.00000	0.00000	0.00000
C++	0.35074	0.11098	0.49058	0.24451	0.23695	0.22880	0.27268	0.00000	0.00000	0.00000	0.00000	0.00000
DELPHI	0.07380	0.08451	0.09229	0.07885	0.08348	0.08615	0.44962	0.00000	0.00000	0.00000	0.00000	0.00000
JAVA	0.31526	0.40331	0.26304	0.29556	0.34187	0.43115	0.16614	0.00000	0.00000	0.00000	0.00000	0.00000
AMAÇ	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

Şekil 5.5. (Devam) Ağırlıklandırılmamış matris

Super Decisions Main Window: YazılımDiliSecimi.mod: Weighted Super Matrix

	1.1.AKA~	1.2.KIT~	1.3.ÖGR~	2.1.YAS~	2.2.END~	2.3.BIL~	3.1.SIS~	3.2.ISL~	3.3.AÇI~	4.1.UY.~	4.2.HAT~	5.1.TEM~	5.2.KOR~
1.1.AKA~	0.00000	0.15436	0.09241	0.08071	0.00000	0.00000	0.00000	0.00000	0.00000	0.11209	0.00000	0.07790	0.50096
1.2.KIT~	0.14241	0.00000	0.11643	0.06510	0.18205	0.00000	0.00000	0.00000	0.00000	0.13451	0.00000	0.40829	0.00000
1.3.ÖGR~	0.04450	0.00000	0.00000	0.05105	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2.1.YAS~	0.03348	0.00000	0.00000	0.00000	0.08202	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2.2.END~	0.20524	0.00000	0.00000	0.14192	0.00000	0.00000	0.00000	0.00000	0.00000	0.28133	0.00000	0.00000	0.33122
2.3.BIL~	0.05243	0.00000	0.00000	0.14192	0.18046	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3.1.SIS~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3.2.ISL~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3.3.AÇI~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
4.1.UY.~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.22369	0.00000	0.00000
4.2.HAT~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.17012	0.40708	0.00000	0.00000
5.1.TEM~	0.00000	0.15373	0.20798	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
5.2.KOR~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
5.3.ILE~	0.00000	0.00000	0.00000	0.13970	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
6.1.WEB~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
6.2.SPE~	0.00000	0.00000	0.00000	0.00000	0.20440	0.00000	0.00000	0.75000	0.00000	0.00000	0.00000	0.00000	0.00000
7.1.IFA~	0.00000	0.13664	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.28345	0.00000
7.2.ÖGR~	0.00000	0.12421	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.06749	0.00000
8.1.DES~	0.23124	0.15409	0.05836	0.20523	0.23000	0.00000	0.00000	0.00000	0.00000	0.11039	0.00000	0.00000	0.00000
8.2.EGI~	0.16058	0.16950	0.37943	0.04348	0.00000	0.00000	0.00000	0.00000	0.00000	0.09199	0.00000	0.00000	0.00000
C#	0.02523	0.02652	0.02507	0.04676	0.03155	0.31073	0.31209	0.03374	0.22403	0.03718	0.14652	0.06873	0.06930
C++	0.03893	0.02963	0.05164	0.02913	0.03034	0.21968	0.21814	0.06835	0.17233	0.00881	0.05205	0.04493	0.02806
DELPHI	0.00856	0.00881	0.01243	0.00917	0.01090	0.08000	0.13308	0.01725	0.08528	0.04115	0.11400	0.01472	0.02531
JAVA	0.05741	0.04250	0.05625	0.04586	0.04828	0.38959	0.33669	0.13066	0.51836	0.01244	0.05664	0.03450	0.04516
AMAÇ	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

Şekil 5.6. Ağırlıklandırılmış matris

	5.3.ILE~	6.1.WEB~	6.2.SPE~	7.1.IFA~	7.2.ÖGR~	8.1.DES~	8.2.EGI~	C#	C++	DELPHI	JAVA	AMAÇ
1.1.AKA~	0.00000	0.00000	0.00000	0.39759	0.27478	0.00000	0.00000	0.04611	0.05849	0.04171	0.06821	0.04635
1.2.KIT~	0.00000	0.21306	0.00000	0.36144	0.13739	0.00000	0.00000	0.06561	0.05001	0.07614	0.02940	0.04171
1.3.ÖGR~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.03241	0.03563	0.02628	0.04652	0.05407
2.1.YAS~	0.00000	0.00000	0.00000	0.00000	0.19500	0.00000	0.00000	0.05663	0.06484	0.03429	0.03507	0.05742
2.2.END~	0.00000	0.14022	0.00000	0.00000	0.00000	0.00000	0.00000	0.10146	0.09304	0.11028	0.07319	0.05931
2.3.BIL~	0.00000	0.11685	0.00000	0.00000	0.00000	0.00000	0.00000	0.03952	0.03973	0.05305	0.08935	0.03142
3.1.SIS~	0.00000	0.21088	0.00000	0.00000	0.00000	0.00000	0.00000	0.05909	0.03854	0.05166	0.01562	0.07130
3.2.ISL~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.04722	0.06251	0.03780	0.06534	0.04285
3.3.AÇI~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.02641	0.03168	0.04326	0.05176	0.04944
4.1.UY.~	0.31791	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.03564	0.02970	0.06210	0.04968	0.03769
4.2.HAT~	0.31791	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.05346	0.05940	0.02700	0.03943	0.09044
5.1.TEM~	0.00000	0.00000	0.00000	0.00000	0.26199	0.00000	0.00000	0.03238	0.02020	0.02907	0.03094	0.03859
5.2.KOR~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.03238	0.02020	0.02747	0.01027	0.02929
5.3.ILE~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.01199	0.03635	0.02020	0.03553	0.02572
6.1.WEB~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.10533	0.02107	0.10557	0.11797	0.07615
6.2.SPE~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.04213	0.12640	0.04189	0.02949	0.08376
7.1.IFA~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.03044	0.03044	0.06088	0.03653	0.04715
7.2.ÖGR~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.06088	0.06088	0.03044	0.05479	0.02947
8.1.DES~	0.00000	0.19781	0.00000	0.00000	0.00000	0.00000	0.00000	0.04030	0.08636	0.09068	0.10363	0.06592
8.2.EGI~	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.08060	0.03454	0.03023	0.01727	0.02198
C#	0.09476	0.04862	0.15409	0.09183	0.04419	0.25390	0.11156	0.00000	0.00000	0.00000	0.00000	0.00000
C++	0.12773	0.01345	0.49058	0.05892	0.03101	0.22880	0.27268	0.00000	0.00000	0.00000	0.00000	0.00000
DELPHI	0.02688	0.01024	0.09229	0.01900	0.01092	0.08615	0.44962	0.00000	0.00000	0.00000	0.00000	0.00000
JAVA	0.11481	0.04887	0.26304	0.07122	0.04473	0.43115	0.16614	0.00000	0.00000	0.00000	0.00000	0.00000
AMAÇ	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

Şekil 5.6. (Devam) Ağırlıklandırılmış matris

Super Decisions Main Window: YazılımDiliSecimi.mod: Limit Matrix

	1.1.AKA~	1.2.KIT~	1.3.ÖGR~	2.1.YAS~	2.2.END~	2.3.BIL~	3.1.SIS~	3.2.ISL~	3.3.AÇI~	4.1.UY.~	4.2.HAT~	5.1.TEM~	5.2.KOR~
1.1.AKA~	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121
1.2.KIT~	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542
1.3.ÖGR~	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562
2.1.YAS~	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726
2.2.END~	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786
2.3.BIL~	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877
3.1.SIS~	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742
3.2.ISL~	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737
3.3.AÇI~	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207
4.1.UY.~	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578
4.2.HAT~	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828
5.1.TEM~	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097
5.2.KOR~	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649
5.3.ILE~	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240
6.1.WEB~	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696
6.2.SPE~	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364
7.1.IFA~	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060
7.2.ÖGR~	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827
8.1.DES~	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910
8.2.EGI~	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428
C#	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525
C++	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356
DELPHI	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731
JAVA	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411
AMAÇ	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

Şekil 5.7. Limit matris

	5.3.ILE~	6.1.WEB~	6.2.SPE~	7.1.IFA~	7.2.ÖGR~	8.1.DES~	8.2.EGI~	C#	C++	DELPHI	JAVA	AMAÇ
1.1.AKA~	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121	0.06121
1.2.KIT~	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542	0.07542
1.3.ÖGR~	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562	0.01562
2.1.YAS~	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726	0.02726
2.2.END~	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786	0.05786
2.3.BIL~	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877	0.03877
3.1.SIS~	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742	0.01742
3.2.ISL~	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737	0.01737
3.3.AÇI~	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207	0.01207
4.1.UY.~	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578	0.02578
4.2.HAT~	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828	0.03828
5.1.TEM~	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097	0.03097
5.2.KOR~	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649	0.00649
5.3.ILE~	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240	0.01240
6.1.WEB~	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696	0.02696
6.2.SPE~	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364	0.04364
7.1.IFA~	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060	0.03060
7.2.ÖGR~	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827	0.02827
8.1.DES~	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910	0.07910
8.2.EGI~	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428	0.04428
C#	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525	0.07525
C++	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356	0.08356
DELPHI	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731	0.04731
JAVA	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411	0.10411
AMAÇ	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000

Şekil 5.7. (Devam) Limit matris

5.3.4. ANP sonuçlarının analizi

ANP çözümü sonunda elde ettiğimiz sonuçlar aşağıdaki gibidir. Buradaki "Limiting" değeri kriterlerin ve altenatiflerin limit matristeki değerleridir.

Icon	Name	Normalized by Cluster	Limiting
No Icon	1.1.AKADEMIK KABUL	0.40202	0.061206
No Icon	1.2.KITAPLARIN BULUNMASI	0.49540	0.075422
No Icon	1.3.ÖGRENCI/ AKADEMIK VERSIYONU	0.10258	0.015618
No Icon	2.1.YASAM DÖNGÜSÜNDEKI SAFHASI	0.22002	0.027260
No Icon	2.2.ENDÜSTRIDE KABULÜ	0.46701	0.057861
No Icon	2.3.BILENLERE OLAN TALEP	0.31296	0.038775
No Icon	3.1.SISTEM GEREKSIMLERI	0.37181	0.017423
No Icon	3.2.ISLETIM SISTEMI BAGIMSIZLIGI	0.37061	0.017367
No Icon	3.3.AÇIK KAYNAK	0.25758	0.012070
No Icon	4.1.UY. GEL. ORTAMINI KUL. KOLAYLIGI	0.40243	0.025780
No Icon	4.2.HATA BULMA KOLAYLIKLARI	0.59757	0.038281
No Icon	5.1.TEMEL KAV. ÖGRENME KOLAYLIGI	0.62108	0.030970
No Icon	5.2.KORUMALI KOD ÜRETIMI	0.13019	0.006492
No Icon	5.3.ILERI ÖZELLIKLER	0.24873	0.012403
No Icon	6.1.WEB TABANLI UY. GELISTIRME	0.38193	0.026964
No Icon	6.2.SPESIFIK UY. DESTEKLEME	0.61807	0.043636
No Icon	7.1.IFADELERI ANLAMA VE YOR. YÖNT.	0.51979	0.030600
No Icon	7.2.ÖGRENME YAKLASIMI	0.48021	0.028270
No Icon	8.1.DESTEGE ULASABILME	0.64112	0.079097
No Icon	8.2.EGITMENLERE GEREKEN EGITIM	0.35888	0.044276
No Icon	C#	0.24256	0.075249
No Icon	C++	0.26934	0.083557
No Icon	DELPHI	0.15249	0.047307
No Icon	JAVA	0.33560	0.104113

Şekil 5.8. ANP çözümüne göre kriterlerin ve alternatiflerin limit öncelik değerleri

Şekil 5.8'deki ANP sonuçlarını incelediğimizde her bir kriterin aldığı limit öncelik değerine (limiting) bakarak kriterler arasında elde ettiğimiz sıralama Çizelge 5.4.'teki gibidir.

Çizelge 5.4. ANP çözümüne göre kriterler arasındaki sıralama

Sıralama	**Kriterler**
1	8.1. Desteğe Ulaşabilme
2	1.2. Kitapların Bulunması
3	1.1. Akademik Kabul
4	2.2. Endüstride Kabulü
5	8.2. Eğitmenlere Gereken Eğitim
6	6.2. Spesifik Uygulamaları Destekleme
7	2.3. Bilenlere Talep
8	4.2. Hata Bulma Kolaylıkları
9	5.1. Temel Kavramları Öğr. Kolaylığı
10	7.1. İfadeleri Anlama ve Yor. Yöntemi
11	7.2. Öğrenme Yaklaşımı
12	2.1. Yaşam Döngüsündeki Safhası
13	6.1. Web Tabanlı Uygulama Geliştirme
14	4.1. Uygulama Gel. Ortamını Kul.Kolaylığı
15	3.1. Sistem Gereksinimleri
16	3.2. İşletim Sistemi Bağımsızlığı
17	1.3.Öğrenci/ Akademik Versiyonu
18	5.3. İleri Özellikler
19	3.3. Açık Kaynak
20	5.2. Korumalı Kod Üretimi

Çizelge 5.4'e göre karar sürecimizde en çok etkili olan kriterin *desteğe ulaşabilme* olduğunu görüyoruz. En çok etkisi olan ikinci kriterin *kitapların bulunması*, üçüncü kriterin ise *akademik kabul* olduğunu görüyoruz. Hemen ardından *endüstride kabul*

kriteri gelmektedir. Çalışmafmızda akademisyenlerin ağırlıkta olduğu bir uzman grubuyla çalışıldığı için aslında böyle bir sıralama hiç de sürpriz sayılmamaktadır. Karar sürecine etkisi en zayıf kriterin ise *korumalı kod üretimi* olduğu görülmektedir. *Açık kaynak* ve *ileri özellikler* hemen hemen aynı öncelik puanına sahip olup sırasıyla etkisi en zayıf olan ikinci ve üçüncü kriterlerdir.

Şekil 5.8'deki ANP sonuçlarına göre her bir alternatifin aldığı limit öncelik değerlerinin normalize edilmiş halinin aldığı puanlar (Normalized by cluster) ve bu puanlara göre seçenekler arasında elde ettiğimiz sıralama aşağıdaki gibidir.

Çizelge 5.5. ANP çözümüne göre yazılım dilleri arasındaki sıralama

1	Java	%33,6
2	C++	%26,9
3	C#	%24,2
4	Delphi	%15,2

Belirlemiş olduğumuz kriterlere göre alternatiflerin aldıkları değerleri incelediğimizde Java'nın diğer yazılım dillerine göre çok daha iyi durumda olduğu görülmektedir. Modelimize göre öncelikli kriterlerde daha iyi olması Java'yı ilk sıraya yerleştirmiştir. İkinci sırada ise C++ yazılım dili vardır. Hemen peşinden C# dili ve son sırada ise Delphi dili gelmektedir.

Şekil 5.8'deki limit önceliklerinden faydalanılarak karar sürecinde etkili olan kriterin önem derecesi belirlenmiştir. Aynı limit öncelikleri kullanılarak kriter kümelerinin öncelikleri de tespit edilebilir. Kriter kümesini oluşturan tüm kriterlerin limit öncelikleri toplamı o kümenin önem derecesini gösterir. Böylelikle kararda hangi kriter kümesinin daha etkin olduğu belirlenebilir.

Modelimizde her bir kriter kümesinin içinde bulunan kriterlerin öncelik değerleri Şekil 5.8'den belirlenip toplanarak Çizelge 5.6 elde edilmiştir.

Çizelge 5.6. ANP çözümüne göre kriter kümeleri arasındaki sıralama

Sıralama	**Kriter Kümeleri**	**Toplam Puanı**
1	Akademik Kabulü	0,1522
2	Endüstride Yaygınlığı	0,1239
3	Dil Desteği ve Eğitim	0,1234
4	Dilin Amacı	0,0706
5	Kullanım Kolaylığı	0,0641
6	Dil Metodolojisi	0,0589
7	Pedagojik Özellikleri	0,0499
8	Yazılım Özellikleri	0,0469

Çizelge 5.5'e göre *akademik kabulü* kriter kümesinin ANP ile yazılım dili seçimimizde en etkili kriter kümesi olduğu, onu *endüstride yaygınlığı* ve *dil desteği ve eğitim* kriter kümelerinin izlediği görülmektedir. Bu üç kriter kümesi karar sürecimize en çok etki eden kriter kümeleridir. *Yazılım özellikleri* kriter kümesinin ise karar sürecimizi en az etkileyen kriter kümesi olduğu görülmektedir.

5.4. ANP ile AHP Sonuçlarının Karşılaştırılması

Bu bölümde her iki yöntemden elde edilen sonuçlara göre kriterler arasındaki sıralama Spearman'nın sıra korelasyon katsayısından faydalanılarak, kriter kümeleri ve alternatifler için elde edilen değerler ise grafiksel olarak karşılaştırılmış ve değerlendirilmişlerdir.

5.4.1. Kriterler için sıra korelasyonu testi

Spearman sıra korelasyonu, aşırı uç değerlerin fazla etkisinde kalmayan ve çok genel ana kitle dağılımları için geçerli sınamalara temel olabilecek bir ilişki ölçüsü olup, sıra sayılarının kullanılmasıyla elde edilir. Kriterler arasında AHP ve ANP çözümüne göre elde edilen sıralamadan (bkz. Çizelge 5.1 ve Çizelge 5.4) Spearman'ın sıra korelasyon katsayısı hesaplanmıştır. Analizde SPSS istatistik yazılımı kullanılmıştır.

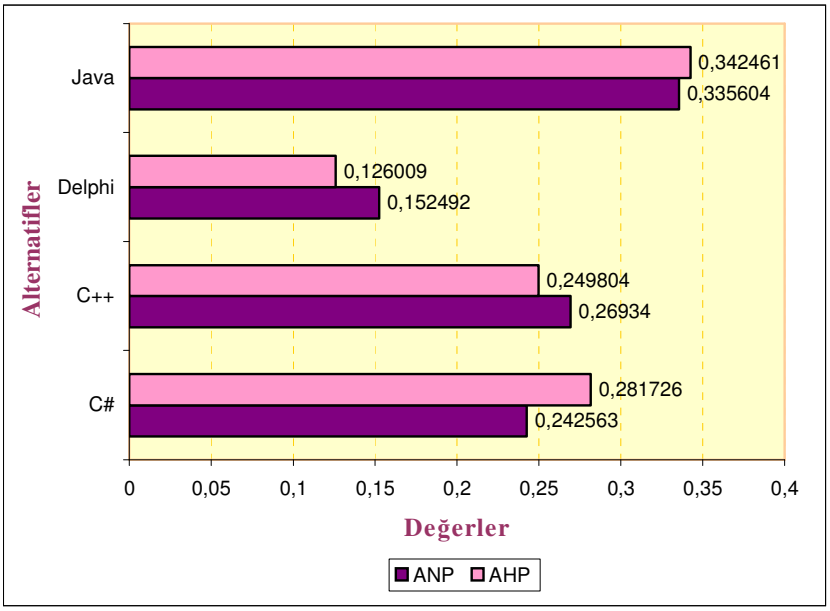
Çizelge 5.7. Spearman'nın sıra korelasyonu testi

			VAR00001	VAR00002
Spearman's rho	VAR00001	Correlation Coefficient	1,000	,623**
		Sig. (1-tailed)	.	,002
		N	20	20
	VAR00002	Correlation Coefficient	,623**	1,000
		Sig. (1-tailed)	,002	.
		N	20	20

Spearman'nın sıra korelasyon katsayısı 0,623 olarak hesaplanmış olup kriterler arasındaki sıralamanın her iki yöntem için genel olarak çok farklı yapıda olmadığını göstermiştir.

5.4.2. Alternatiflerin ve kriter kümelerinin AHP ve ANP çözümlerine göre karşılaştırılması

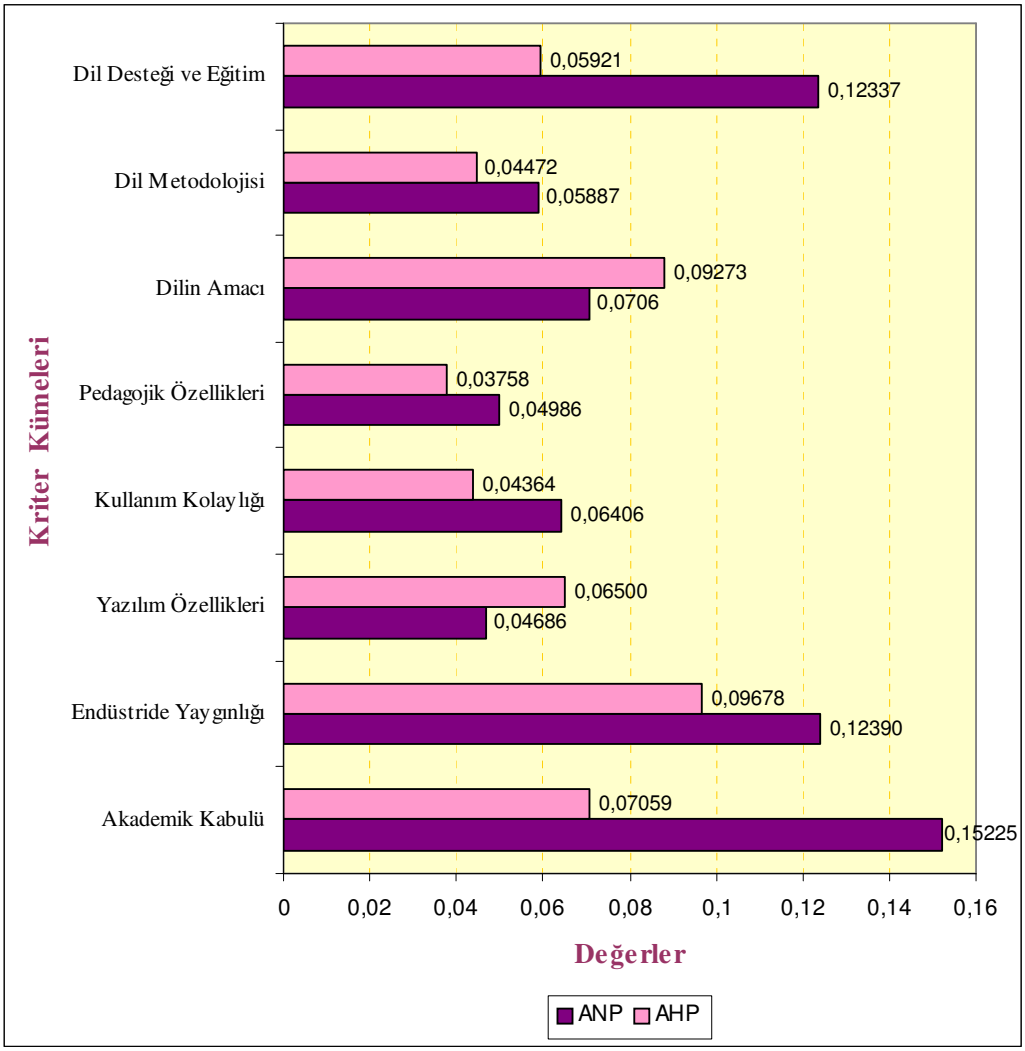
Yazılım dili seçimi için alternatifler arasında AHP ve ANP çözümüne göre elde ettiğimiz sonuçlar Şekil 5.9'da karşılaştırılmıştır.



Şekil 5.9. AHP ve ANP çözümlerine göre alternatiflerin karşılaştırılması

Hem AHP ve hem de ANP'ye göre en yüksek puanı alan dil Java olmuştur. İkinci sırada ise AHP tekniğine göre 0,2817 değeriyle C# gelirken, ANP tekniğine göre 0,2693 değeriyle ile C++ gelmiştir. Üçüncü sırada ise AHP tekniğine göre C++ yer alırken,ANP tekniğine göre C# yer almıştır. Dördüncü yani son sırada hem AHP hem de ANP tekniğine göre Delphi yer almıştır. Kısaca AHP tekniğine göre ikinci ve üçüncü yazılım dilleri arasındaki sıralama (C# ve C++), çalışma ANP tekniğine göre yapıldığında tam tersine dönmüştür.

Yazılım dili seçiminde kullandığımız kriter kümelerinin AHP ve ANP metodundan elde edilen sonuçlara göre karşılaştırılmaları Sekil 5.10'da görülmektedir.



Şekil 5.10. Kriter kümelerinin AHP ve ANP sonuçlarına göre karşılaştırılması

Buna göre *akademik kabulü*, *endüstride yaygınlığı* ve *dil desteği ve eğitim* iki metoda göre alınan sonuçlar arasındaki farkın en çok olduğu kriter kümeleridir.

AHP metoduna göre karar sürecine en çok etkisi olan kriter kümesinin 0,09678 değeriyle *endüstride yaygınlığı* olduğunu görüyoruz. İkinci sırada ise 0,09273 değeriyle *dilin amacı* kriter kümesi yer almaktadır. Üçüncü sıradaki kriter kümesi ise 0,07059 değeri ile *akademik kabulü*dür. AHP ile karar sürecine etkisi en az olan kriter kümesi ise 0,03758 değeri ile *pedagojik özellikleri*dir. Karar sürecine etkisi az olan ikinci kriter kümesi ise 0,04364 değeri ile *kullanım kolaylığı*dır.

ANP metoduna göre karar sürecine en çok etkisi olan kriter kümesinin 0,15225 değeri ile *akademik kabulü* olduğunu görüyoruz. 0,12390 değeri ile *endüstride yaygınlığı* ve 0,12337 değeri ile *dil desteği ve eğitim* kriter kümeleri hemen hemen aynı puanı almış olup ikinciliği paylaşmaktadırlar. Hemen peşlerinden 0,0706 değeri ile *dilin amacı* kriter kümesi gelmektedir. ANP ile karar sürecine etkisi en az olan kriter kümesi ise 0,04686 değeri ile *yazılım özellikleri*dir. *Pedagojik özellikleri* ise ANP ile karar sürecine etkisi en az olan ikinci kriter kümesidir.

Genel olarak baktığımızda *dilin amacı* ve *yazılım özellikleri* dışındaki kriter kümelerinde ANP metoduna göre hesaplanan değerlerin AHP metodu ile hesaplanandan daha yüksek olduğu görülmektedir. Bunda kriterler arasındaki etkileşimin büyük etkisi vardır. Kriter arasındaki etkileşimin en yoğun olduğu *akademik kabulü* kriter kümesinde AHP ve ANP metoduna göre hesaplanmış değerler arasında dikkat çekici bir fark görülmektedir. Bu çalışmada kriterler arasındaki bağımlılığı dikkate almanın önemi ve gerekliliği görülmektedir. Dolayısıyla sağlıklı ve doğru kararlar verebilmek için ANP sonuçları esas alınmalıdır.

6. SONUÇ VE ÖNERİLER

Dünyada yazılımın önemi giderek artarken yazılım endüstrisi, stratejik bir sektör olma yolunda ilerlemektedir. Bunu kavrayabilen ülkeler, kendi yazılım endüstrilerinin gelişmesi için ulusal boyutta stratejiler geliştirerek uzun vadeli uygulama planları hazırladılar. Bu ülkeler aynı zamanda özel teşvikler sunarak ve en önemlisi de bütün bunları kararlılıkla sürdürerek yol aldılar. Bu ülkelere örnek olarak, Hindistan, İrlanda, İsviçre, İsrail gösterilmektedir. Türkiye de Bilişim Teknolojisi üretmelidir. Dünya ile rekabet eden bir ekonomi ancak yoğun ARGE ve Bilişim Teknolojisi üretimiyle mümkündür. Başta yazılım olmak üzere, tüm bilişim sektöründe geniş bir yelpazede ve genelde kritik teknolojilerde üretim yapılmalıdır. Bilişim Teknolojisi sektöründe ARGE ve üretim alanlarında gerekli teşvik ve planlama yapılmalıdır [31]. Bilişim sektörü için önemli bir konu olan insan gücünün verimlilik ve etkinliğini artıracak çalışmalar ve yatırımlar yapılarak sağlam bir alt yapı oluşturulmalıdır. Bilişim konusunda bilimsel araştırma ve geliştirme çalışmaları ile yazılım üretimi desteklenmeli, kalitenin gelişimi için gerekenler yapılmalıdır.

Yazılım geliştiren firmalar, etkinliklerini arttırmak ve başarılı bir yazılım ortaya koyabilmek için yazılım dillerini en etkin şekilde kullanmak durumundadır. Birçok firmada yazılım dili seçimi üst yönetim tarafından gelişigüzel yapılmaktadır. Üst yönetimdekiler genellikle yazılım dilinin uygun olup olmadığına bakmadan her yere, her duruma uygulamasını isterler ve maalesef işin içinden çıkamazlar. Tek bir yazılım dilinin tüm uygulamalar için en iyisi, en ideali olduğunu düşünmek ve bu konuda ısrarcı olmak işleri zora sokabilir.

Yazılım dilleri insanlara benzetilebilir. Hepsinin bir gelişim süreci, geçmişi olduğu gibi diğer dillere göre üstünlükleri ve zayıf yönleri vardır. Asıl mesele bunları doğru analiz edip uygun yerde uygun yazılım dilini kullanabilmektir. Yanlış yazılım dili seçimi aslında birçok projenin zamanında tamamlanmamasının ya da ihtiyaçları tam olarak karşılayamamasının altında yatan en önemli etkenlerden biridir.

Bu tez çalışmasında literatür araştırması ile yazılım dili seçimi için yirmi adet kriter belirlenmiş daha sonra bu kriterler sekiz grupta toplanmıştır. Belirlenen iç ve dış bağımlılıklara göre network yapısı ve ANP modeli oluşturulmuştur. ANP modeline göre oluşturduğumuz anketler, çalışmanın daha objektif olması açısından her biri farklı bir üniversitenin bilgisayar mühendisliğinde çalışan akademisyenlerden ve bilgisayar mühendislerinden oluşan uzman bir ekibe doldurulmuştur. Bu anket sonuçlarının geometrik ortalaması alınarak nihai çözüm için gerekli veriler elde edilmiştir. Bu veriler Super Decisions programına elle girilerek ağırlıklandırılmış matris, ağırlıklandırılmamış matris ve limit matris elde edilmiştir. Limit matris sonuçlarına bakılarak alternatifler ve kriterler arasında bir sıralama elde edilmiştir. Söz konusu nesne yönelimli yazılım dilleri arasında göreli önemlerine göre bir değerlendirme yapılmıştır. Kriter arasındaki sıralamadan faydalanılarak kriter kümeleri arasındaki sıralama elde edilmiştir ve karar sürecinde en çok ve en az etkili olan kriter kümeleri belirlenmiştir. Anketlerden elde ettiğimiz veriler kullanılarak oluşturduğumuz AHP modeli de çözümlenmiş, alternatifler ve kriterler arasındaki sıralama verilmiştir. Çalışmanın sonunda alternatifler, kriterler ve kriter kümeleri arasında her iki metoda göre sıralama yapılmış, elde edilen sonuçlar karşılaştırılmıştır. Bu tip bir çalışmada AHP sonuçlarına bakarak seçim yapmanın çok sağlıklı olmayacağı, ANP metodunun kriterler arası bağımlılığı da gözönünde tutması bakımından daha sağlıklı ve gerçekçi kararlar almaya yardımcı olduğu tespit edilmiştir.

Sonuç olarak bu tez çalışması ile yazılım geliştirenler, öğretenler ya da öğrenenler için yazılım dili seçimine analitik bakış açısıyla nasıl yaklaşılacağı gösterilmeye çalışılmıştır. Çalışma pek çok unsuru dikkate almış ve titizlikle yürütülmüştür. Ayrıca yazılım dili seçimi için literatürde ANP metodu ile yapılmış ilk çalışmadır. Bu çalışmada kullanılan nitel kriterlere yenileri eklenerek ya da ihtiyaca göre maliyet, performans gibi nicel kriterler de eklenerek genişletilebilir. Yazılımda rekabet avantajı yaratma ve sürdürebilme ancak doğru ürünü, doğru şekilde, doğru müşteriye ve doğru yazılım dili ile üretmekle sağlanabilir.

KAYNAKLAR

1. Islıkçı, C., Envarlı, B., Yılmaz, V. Tekbulut T. ve Diğerleri, "Türkiye'de Bilişim Sektörünün Gelişimi", Sektörün Ana Raporu, ***Türkiye Bilişim Şurası***, Ankara, 15-18, 43-44 (2002).

2. Golub, A. L. , "Decision Analysis An Integrated Approach", ***John Wiley & Sons Inc.***, America, 2-12, 230-231 (1997).

3. Hoy, W. K. ve Tarter, C. J. , "Administrators Solving the Problems of Practice", Second Ed., ***Pearson Education Inc.***, USA, 15 (2004).

4. Evren, Ramazan ve F. Ülengin, "Yönetimde Karar Verme**"**, ***İTÜ Matbaası***, İstanbul, 5-6, 48, 59 (1992).

5. Tekeş, M., "Çok Ölçütlü Karar Verme Yöntemleri ve Türk Silahlı Kuvvetleri'nde Kullanılan Tabancaların Bulanık Uygunluk İndeksli Analitik Hiyerarşi Prosesi İle Karşılaştırılması", Yüksek Lisans Tezi, ***İTÜ Fen Bilimleri Enstitüsü***, İstanbul, 4-5,12-14 (2002).

6. Kıvrak, E., "Karar Verme Çok Kriterli Yaklaşım ve Analitik Hiyerarşi Yöntemi", Yüksek Lisans Tezi, ***Başkent Üniversitesi Sosyal Bilimler Enstitüsü***, Ankara, 35-39, 43 (2001).

7. Aytürk, S., "Askeri Savunma Sistemlerinde Analitik Hiyerarşi ve Analitik Şebeke Prosesi ile Hafif Makineli Tüfek Seçimi", Yüksek Lisans Tezi, ***Gazi Üniversitesi Fen Bilimleri Enstitüsü***, Ankara, 5-9, 24-25,30(2006).

8. Ossadnik, W., Lange, O., "AHP-based evaluation of AHP-Software", ***European Journal of Operational Research*** , 118, 578-588 (1999).

9. Zhu, X., Dale, A. P., "JavaAHP: A Web-Based Decision Analysis Tool For Natural Resource And Environmental Management", ***Environmental Modelling & Software*** , 16: 251–262 (2001).

10. Gönenç, Dinçer S., "Analysis Of The Factor That Affect The Motivation And Job Performance Of The Software Development Teams", Yüksek Lisans Tezi, ***ODTÜ Fen Bilimleri Enstitüsü***, Ankara,1-3 (2001).

11. Lai, V. S., Wong, B.K., Cheung W., " Group decision making in a multiple criteria environment: A case using the AHP in software selection", ***European Journal of Operational Research***, 134-144(2002).

12. Yücel D., Erkut H., "Bilişim Teknolojilerinin Çalışma Yaşam Kalitesi Üzerine Etkisi", ***İTÜ Dergisi,*** 2(2): 49-59 (2003).

13. Lee, Y., Kozar, K. A. And Smith, J. M., "Investigating the effect of website quality on e-business success:An analytic hierarchy process (AHP) approach" , ***Decision Support Systems***, 1-6 (2005).

14. Wei,C., Chien, C., Wang M. J.," An AHP-based approach to ERP system selection", ***Production Economics*** , 47-62(2005).

15. Başlıgil, H., "The Fuzzy Analytic Hierarchy Process For Software Selection Problems", ***Sigma Mühendislik ve Fen Bilimleri Dergisi,*** 3: 24-33(2005).

16. Ngai, E.W.T. , Chan E.W.C., "Evaluation of Knowledge Management Tools Using AHP", ***Expert Systems with Applications*** , 29: 889–899(2005).

17. Repiso, L. R., Setchi, R. ve Salmeron, J. L., "Modelling IT projects success: Emerging methodologies reviewed", ***Technovation*** (2007).

18. Erikan, L., "Hava Kuvvetleri Komutanlığında Aday Seçiminde AHP ile Etkin Karar Verme", Yüksek Lisans Tezi, ***İTÜ Fen Bilimleri Enstitüsü***, İstanbul, 64-66 (2002).

19. Rouyandegh, B. D., "DEA/AHP sıralı metodu ile İran Amir Kabir Üniversitesi Fakültelerinin Etkinlik Ölçümü ", Yüksek Lisans Tezi, ***Gazi Üniversitesi Fen Bilimleri Enstitüsü***, Ankara, 18-19 (2004).

20. Dağdeviren, M., "Analitik Hiyerarşi Prosesi ile Yeni Bir Analitik İş Değerlendirme Tekniğinin Geliştirilmesi", Yüksek Lisans Tezi, ***Gazi Üniversitesi Fen Bilimleri Enstitüsü***, Ankara,47-63 (2002).

21. Saaty, T., "Decision Making For Leaders", ***RWS Publications***, USA, 1-30 (1990).

22. Saaty, T., "The Analitic Hierarchy Process", ***McGraw-Hill International Book Company***, New York, 1-72, 230-345 (1980).

23. Kuruüzüm A. ve Atsan N., **"**Analitik Hiyerarşi Yöntemi Ve İşletmecilik Alanındaki Uygulamaları", ***Akdeniz İ.İ.B.F. Dergisi,*** Antalya,1: 83-105(2001).

24. Dağdeviren , M. , Eraslan, E. Ve Kurt, M., **"**Çalışanların Toplam İş Yükü Seviyelerinin Belirlenmesine Yönelik Bir Model ve Uygulaması", ***Gazi Üniversitesi Müh. Mim. Fak. Der.***, Ankara, 20(4): 518-519 (2005).

25. Saaty, T.L., "Decision Making with Dependence and Feedback: The Analytic Network Process", ***RWS Publications***, Pittsburgh, 76-77(1996).

26. Büyükyazıcı, M., "Analitik Ağ Süreci", Yüksek Lisans Tezi, ***Hacettepe Üniversitesi Fen Bilimler Enstitüsü***, Ankara, 43,44 (2000).

27. Ertiketlioğlu, A., "Askeri Lojistik Politikaların Belirlenmesinde Analitik Şebeke Yönetimi", ***İTÜ Fen Bilimleri Enstitüsü***, İstanbul, 21, 22 (2000).

28. Staugaard, A. C., "Information Systems Programming with Java", ***Pearson Prentice Hall***, USA, 9-12 (2004).

29. Deitel H. M., Deitel P. J., " C ve C++", ***Prentice Hall***, Sistem Yayıncılık , USA, 9-13 (2004).

30. Baransel, C., Mumcuoğlu, A., "Web Tabanlı, Üç Katmanlı Yazılım Mimarileri: ORACLE, UML ve EJB Ile Sistem Modelleme, Tasarım ve Gerçekleştirim", ***SAS Yayınları***, 2-13 (2003).

31. İnternet: Devlet Planlama Teşkilatı http://www.bilgitoplumu.gov.tr/btstrateji/Strateji_Belgesi.pdf (2006).

32. Parker K. R., Chao J. T., *Ottaway T. A., Chang J.,* "A Formal Language Selection Process for Introductory Programming Courses", ***Journal of Information Technology Education,*** USA, 5, 135-142 (2006).

33. Roberts, E. Resources to support the use of java in introductory computer science. ***ACM SIGCSE Bulletin****, 36* (1): 233-234 (2004).

34. Lee, P.A., & Stroud, R.J. C++ as an introductory programming language. In M. Woodman (Ed.), ***Programming Language Choice: Practice and Experience*** London: International Thomson 63-82 (1996).

35. King, K.N. "The evolution of the programming languages course", ***ACM SIGCSE Bulletin*****,** *24* (1): 213-219 (1992).

36. Howland, J.E. "It's all in the language: Yet another look at the choice of programming language for teaching computer science" ***Journal of Computing in Small Colleges****, 12* (4): 58-74 (1997).

37. Raadt, M., Watson, R., & Toleman, M. Introductory programming languages at Australian universities at the beginning of the twenty first century. ***Journal of Research and Practice in Information Technology****, 35* (3): 163-167(2003).

38. Eisenstadt, M., & Lewis, M.W. Errors in an interactive programming environment: Causes and cures. In M. Eisenstadt, M.T. Keane, & T. Rajan (Eds.), "Novice programming environments: Explorations in human-computer interaction and artificial intelligence", ***Hillsdale, NJ: Lawrence***, Chapter 5 (1992).

39. Kölling, M., Koch, B., & Rosenberg, J. Requirements for a first year object oriented teaching language., ***ACM SIGCSE Bulletin****, 27* (1): 173-177 (1995).

40. McIver, L., & Conway, D.M. Seven deadly sins of introductory programming language design, "Proceedings of Software Engineering: Education and Practice", ***IEEE Computing Society Press,*** USA, 309-316 (1996).

41. Conway, D. "Criteria and considerations in the selection of a first programming language", Technical Report, ***Department of Computer Science, Monash University***, *93/192* (1993).

42. Howatt, J. W. "A project-based approach to programming language evaluation", ***ACM SIGPLAN Notices,*** 30 (7): 37-40 (1995).

43. Stroustrup B., "Learning Standard C++ as a New Language", ***The C/C++ Users Journal*** , 9-10 (1999).

44. Tharp, A.L. "Selecting the 'right' programming language", ***ACM SIGCSE Bulletin***, 14(1): 151-155 (1982).

45. Emigh, K. L., "The impact of new programming languages on university curriculum" ***Proceedings of ISECON 2001****,* Cincinnati, Ohio*, 18*, 1146-1151 (2001).

46. Güngören B., "C++ ile Nesne Tabanlı Programlama" ,2. baskı, ***Seçkin Yayıncılık San. Ve Tic. A. Ş.***, Ankara, 23-24 (2004).

47. Sethupathi M., Jayaraman R., "Special edition Using C#", ***NITT***, USA, 11-13 (2002).

48. Peköz, N., " Java", ***Pusula Yayıncılık ve İletişim Ltd. Şti***., İstanbul, 10-17 (2003).

49. Köseoğlu K., "Programcılık Mantığı", ***Pusula Yayıncılık***, 19-20 (2004).

EKLER

EK -1 Anket

Uzmanlar tarafından bilgisayar başında tutarlılık oranları tek tek kontrol edilerek doldurulan anket formları aşağıda verilmiştir. Anketler ikili karşılaştırmalar şeklinde hazırlanmış olup üç başlık altında toplanmıştır. Uzmanlara anketler hakkında gerekli açıklamalar yapılmış, nasıl dolduracakları konusunda detaylı bilgi verilmiştir. Aşağıda ikili karşılaştırmaların nasıl yorumlandığını gösteren bir örnek verilmiştir.

Örnek:

1 : eşit 3: biraz daha fazla 5: daha fazla 7:çok daha fazla 9: aşırı derecede fazla

1_Akademik Kabulü	9	8	7	6	5	4	3	2	**1**	2	3	4	5	6	7	8	9	*2_Endüstride Yaygınlık*

Akademik Kabulü ve *Endüstride Yaygınlık* kriter kümelerinin eşit derecede önemli olduğunu düşünen uzman yukarıda görüldüğü gibi 1'i işaretlemiştir.

1_Akademik Kabulü	9	8	7	6	5	4	**3**	2	1	2	3	4	5	6	7	8	9	*2_Endüstride Yaygınlık*

Akademik Kabulü kriter kümesinin *Endüstride Yaygınlık* kriter kümesinden biraz daha fazla önemli olduğunu düşünen uzman yukarıda görüldüğü gibi 3'ü işaretlemiştir.

1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	**7**	8	9	*2_Endüstride Yaygınlık*

Endüstride Yaygınlık kriter kümesinin *Akademik Kabulü* kriter kümesinden çok daha fazla önemli olduğunu düşünen uzman yukarıda görüldüğü gibi 7'yi işaretlemiştir

EK-1 (Devam) Anket

KRİTER KÜMELERİNİN İKİLİ KARŞILAŞTIRMALARI

1_AKADEMİK KABULÜ kriter kümesi ile ilgili ikili küme karşılaştırmaları

1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2_EndüstrideYaygınlık*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
2_EndüstrideYaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
7_Dil Metodolojisi	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
7_Dil Metodolojisi	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
8_Dil Desteği ve Eğitim	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*

2_ENDÜSTRİDE YAYGINLIĞI kriter kümesi ile ilgili ikili küme karşılaştırmaları

1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2_Endüstride Yaygınlık*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
6_Dilin Amacı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
6_Dilin Amacı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
8_Dil Desteği ve Eğitim	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*

EK-1 (Devam) Anket

3_YAZILIM ÖZELLİKLERİ kriter kümesi ile ilgili ikili küme karşılaştırmaları

6_Dilin Amacı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*

4_KULLANIM KOLAYLIĞI kriter kümesi ile ilgili ikili küme karşılaştırmaları

1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2_Endüstride Yaygınlık*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4_Kullanım Kolaylığı*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4_Kullanım Kolaylığı*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
8_Dil Desteği ve Eğitim	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*

5_PEDAGOJİK ÖZELLİKLERİ kriter kümesi ile ilgili ikili küme karşılaştırmaları

1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2_Endüstride Yaygınlığı*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4_Kullanım Kolaylığı*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4_Kullanım Kolaylığı*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
7_Dil Metodolojisi	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*

EK-1 (Devam) Anket

6_DİLİN AMACI kriter kümesi ile ilgili ikili küme karşılaştırmaları

1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2_Endüstride Yaygınlık*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3_Yazılım Öz.*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3_Yazılım Öz.*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
3_Yazılım Özellikleri	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
3_Yazılım Özellikleri	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
8_Dil Desteği ve Eğitim	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*

7_DİL METODOLOJİSİ kriter kümesi ile ilgili ikili küme karşılaştırmaları

1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2_Endüstride Yaygınlık*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*Alternatifler*

EK-1 (Devam) Anket

ALTERNATİFLER için ikili küme karşılaştırmaları

1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2_Endüstride Yaygınlığı*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3_Yazılım Öz.*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4_Kullanım Kolaylığı*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3_Yazılım Öz.*
2_Endüstride Yaygınlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4_Kullanım Kolaylığı*
2_Endüstride Yaygınlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
2_Endüstride Yaygınlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
2_Endüstride Yaygınlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
2_Endüstride Yaygınlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
3_Yazılım Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4_Kullanım Kolaylığı*
3_Yazılım Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
3_Yazılım Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
3_Yazılım Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
3_Yazılım Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
6_Dilin Amacı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
6_Dilin Amacı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
7_Dil Metodolojisi	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*

EK-1 (Devam) Anket

AMAÇ için ikili küme karşılaştırmaları

1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2_Endüstride Yaygınlığı*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3_Yazılım Öz.*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4_Kullanım Kolaylığı*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
1_Akademik Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
2_Endüstride Yaygınlık	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3_Yazılım Öz.*
2_Endüstride Yaygınlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4_Kullanım Kolaylığı*
2_Endüstride Yaygınlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
2_Endüstride Yaygınlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
2_Endüstride Yaygınlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
2_Endüstride Yaygınlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
3_Yazılım Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4_Kullanım Kolaylığı*
3_Yazılım Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
3_Yazılım Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
3_Yazılım Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
3_Yazılım Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5_Pedagojik Öz.*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
4_Kullanım Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6_Dilin Amacı*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
5_Pedagojik Öz.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
6_Dilin Amacı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7_Dil Metodolojisi*
6_Dilin Amacı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*
7_Dil Metodolojisi	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8_Dil Desteği ve Eğitim*

EK-1 (Devam) Anket

KRITERLERİN BAĞLI OLDUKLARI KRITERLER ARASINDAKİ İKİLİ KARŞILAŞTIRMA

1.1.Akademik Kabul kriteri açısından aşağıdaki kriterlerin ve alternatiflerin ikili karşılaştırması :

İç bağımlılık

1.2.Kitapların Bulunması	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.3.Öğrenci Versiyonu*

Dış bağımlılıklar

2.1.Yaşam Dön. Safhası	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2.2.Endüstride Kabulü*
2.1.Yaşam Dön. Safhası	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2.3.Bilenlere Talep*
2.2.Endüstride Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2.3.Bilenlere Talep*

8.1.Desteğe Ulaşabilme	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8.2.Eğitmenlerin Eğitimi*

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

1.2.Kitapların Bulunması kriteri açısından aşağıdaki kriterlerin ve alternatiflerin ikili karşılaştırması :

Dış bağımlılıklar

7.1.İfadeleri Anlama &Yorumlama Yöntemi	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7.2.Öğrenme Yaklaşımı*

8.1.Desteğe Ulaşabilme	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8.2.Eğitmenlerin Eğitimi*

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

EK-1 (Devam) Anket

1.3.Öğrenci/ Akademik Versiyonu kriteri açısından aşağıdaki kriterlerin ve alternatiflerin ikili karşılaştırması :

İç bağımlılık

1.1. Akademik Kabul	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.2.Kitapların Bulunması*

Dış bağımlılıklar

8.1.Desteğe Ulaşabilme	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8.2.Eğitmenlerin Eğitimi*

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

2.1.Yaşam Döngüsündeki Safhası kriteri açısından aşağıdaki kriterlerin ve alternatiflerin ikili karşılaştırması :

İç bağımlılık

2.2.Endüstride Kabulü	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*2.3. Bilenlere Talep*

Dış bağımlılıklar

1.1. Akademik Kabul	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.2.Kitapların Bulunması*
1.1. Akademik Kabul	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.3.Öğrenci Versiyonu*
1.2. Kitapların Bulunması	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.3.Öğrenci Versiyonu*

8.1.Desteğe Ulaşabilme	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8.2.Eğitmenlerin Eğitimi*

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

EK-1 (Devam) Anket

2.2.Endüstride Kabulü kriteri açısından aşağıdaki kriterlerin ve alternatiflerin ikili karşılaştırması :

İç bağımlılık

2.1.Yaşam Dön. Saf.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2.3. Bilenlere Talep*

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

2.3.Bilenlere Talep kriteri açısından aşağıdaki alternatiflerin ikili karşılaştırması :

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

3.1.Sistem Gereksinimleri kriteri açısından aşağıdaki alternatiflerin ikili karşılaştırması :

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

3.2.İşletim Sistemi Bağımsızlığı kriteri açısından aşağıdaki alternatiflerin ikili karşılaştırması :

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

EK-1 (Devam) Anket

3.3. Açık Kaynak kriteri açısından aşağıdaki alternatiflerin ikili karşılaştırması :

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

4.1. Uygulama Gel. Ortamını Kul.Kolaylığı kriteri açısından aşağıdaki kriterlerin ve alternatiflerin ikili karşılaştırması :

Dış bağımlılıklar

1.1. Akademik Kabul	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.2.Kitapların Bulunması*

8.1.Desteğe Ulaşabilme	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8.2.Eğitmenlerin Eğitimi*

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

4.2. Hata Bulma Kolaylıkları kriteri açısından aşağıdaki alternatiflerin ikili karşılaştırması :

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

EK-1 (Devam) Anket

5.1. Temel Kavramları Öğr. Kolaylığı kriteri açısından aşağıdaki kriterlerin ve alternatiflerin ikili karşılaştırması :

Dış bağımlılıklar

1.1. Akademik Kabul	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.2.Kitapların Bulunması*

7.1. İfadeleri Anlama ve Yor. Yöntemi	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7.2. Öğrenme Yaklaşımı*

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

5.2. Korumalı Kod Üretimi kriteri açısından aşağıdaki aşağıdaki alternatiflerin ikili karşılaştırması :

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

5.3. İleri Özellikler kriteri açısından aşağıdaki kriterlerin ve alternatiflerin ikili karşılaştırması :

Dış bağımlılıklar

4.1. Uygulama Gel. Ortamını Kul. Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4.2. Hata Bulma Kolaylıkları*

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

EK-1 (Devam) Anket

6.1. Web Tabanlı Uygulama Geliştirme kriteri açısından aşağıdaki kriterlerin ve alternatiflerin ikili karşılaştırması :

Dış bağımlılıklar

2.2. Endüstride Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2.3. Bilenlere Talep*

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

6.2. Spesifik Uygulamaları Destekleme kriteri açısından aşağıdaki alternatiflerin ikili karşılaştırması :

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

7.1. İfadeleri Anlama ve Yor. Yöntemi kriteri açısından aşağıdaki kriterlerin ve alternatiflerin ikili karşılaştırması :

Dış bağımlılıklar

1.1. Akademik Kabul	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.2.Kitapların Bulunması*

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

EK-1 (Devam) Anket

7.2. Öğrenme Yaklaşımı kriteri açısından aşağıdaki kriterlerin ve alternatiflerin ikili karşılaştırması :

Dış bağımlılıklar

1.1. Akademik Kabul	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.2.Kitapların Bulunması*

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

8.1.Desteğe Ulaşabilme kriteri açısından aşağıdaki alternatiflerin ikili karşılaştırması :

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

8.2. Eğitmenlere Gereken Eğitim kriteri açısından aşağıdaki alternatiflerin ikili karşılaştırması :

Alternatifler

C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*C++*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C#	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*DELPHI*
C++	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*
DELPHI	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*JAVA*

EK-1 (Devam) Anket

KRITER KÜMELERİNİN ELEMANLARININ İKİLİ OLARAK KARŞILAŞTIRILMASI

Tüm alternatifler(C#, C++, Delphi, Java) için ayrı ayrı kriter kümelerinin elemanlarını ikili olarak karşılaştırma :

1.1.Akademik Kabul	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.2.Kitapların Bulunması*
1.1.Akademik Kabul	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.3.Öğrenci/ Akademik Versiyonu*
1.2.Kitapların Bulunması	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.3.Öğrenci/ Akademik Versiyonu*

2.1.Yaşam Döngüsündeki Safhası	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2.2.Endüstride Kabulü*
2.1.Yaşam Döngüsündeki Safhası	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2.3.Bilenlere Talep*
2.2.Endüstride Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2.3.Bilenlere Talep*

3.1.Sistem Gereksinimleri	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3.2.İşletim Sistemi Bağımsızlığı*
3.1.Sistem Gereksinimleri	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3.3.Açık Kaynak*
3.2.İşletim Sistemi Bağımsızlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3.3.Açık Kaynak*

4.1.Uygulama Gel. Ortamını Kul.Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4.2.Hata Bulma Kolaylıkları*

5.1.Temel Kavramları Öğr. Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5.2.Korumalı Kod Üretimi*
5.1.Temel Kavramları Öğr. Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5.3.İleri Özellikler*
5.2.Korumalı Kod Üretimi	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5.3.İleri Özellikler*

6.1.Web Tabanlı Uygulama Geliştirme	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6.2.Spesifik Uygulamaları Destekleme*

7.1.İfadeleri Anlama ve Yorumlama Yön.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7.2.Öğrenme Yaklaşımı*

8.1.Desteğe Ulaşabilme	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8.2.Eğitmenlerin Eğitimi*

EK-1 (Devam) Anket

AMAÇ için kriter kümelerinin elemanlarını ikili olarak karşılaştırma :

1.1.Akademik Kabul	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.2.Kitapların Bulunması*
1.1.Akademik Kabul	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.3.Öğrenci/ Akademik Versiyonu*
1.2.Kitapların Bulunması	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*1.3.Öğrenci/ Akademik Versiyonu*

2.1.Yaşam Döngüsündeki Safhası	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2.2.Endüstride Kabulü*
2.1.Yaşam Döngüsündeki Safhası	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2.3.Bilenlere Talep*
2.2.Endüstride Kabulü	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*2.3.Bilenlere Talep*

3.1.Sistem Gereksinimleri	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3.2.İşletim Sistemi Bağımsızlığı*
3.1.Sistem Gereksinimleri	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3.3.Açık Kaynak*
3.2.İşletim Sistemi Bağımsızlığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*3.3.Açık Kaynak*

4.1.Uygulama Gel. Ortamını Kul.Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*4.2.Hata Bulma Kolaylıkları*

5.1.Temel Kavramları Öğr. Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5.2.Korumalı Kod Üretimi*
5.1.Temel Kavramları Öğr. Kolaylığı	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5.3.İleri Özellikler*
5.2.Korumalı Kod Üretimi	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*5.3.İleri Özellikler*

6.1.Web Tabanlı Uygulama Geliştirme	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*6.2.Spesifik Uygulamaları Destekleme*

7.1.İfadeleri Anlama ve Yorumlama Yön.	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*7.2.Öğrenme Yaklaşımı*

8.1.Desteğe Ulaşabilme	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	*8.2.Eğitmenlerin Eğitimi*

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: ANIK, Zelal
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 08.06.1980 Diyarbakır
Medeni hali	: Bekar
Telefon	: 0 (312) 4422881
e-mail	: zelal_anik@yahoo.com

Eğitim Derece	Eğitim Birimi	Mezuniyet tarihi
Yüksek lisans	Gazi Üniversitesi /Endüstri Mühendisliği	2007
Lisans	Gazi Üniversitesi/Endüstri Mühendisliği	2003
Lise	Yüce Fen Lisesi	1998

İş Deneyimi

Yıl	Yer	Görev
2004	Anatolia Bilgisayar Ltd. Şti.	ERP Danışmanı
2005-2006	Simetri Yazılım A. Ş.	Analist Programcı

Yabancı Dil

İngilizce, Almanca

Hobiler

Resim, Takı Tasarımı, Aerobik, Yürüyüş, Yoga, Satranç