
GAZİ ÜNİVERSİTESİ
Gazi
1926

RADAR VE ELEKTRONİK HARP PROJELERİNDE "GÖREV PLANLAMA" YAZILIMLARI İÇİN YAZILIM ÜRÜN HATTI MİMARİSİ KAPSAMINDA YENİDEN KULLANILABİLİRLİK YÖNTEMLERİNİN UYGULANMASI

Melis BIYIKLI

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

AĞUSTOS 2020

Melis BIYIKLI tarafından hazırlanan "RADAR VE ELEKTRONİK HARP PROJELERİNDE "GÖREV PLANLAMA" YAZILIMLARI İÇİN YAZILIM ÜRÜN HATTI MİMARİSİ KAPSAMINDA YENİDEN KULLANILABİLİRLİK YÖNTEMLERİNİN UYGULANMASI" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilgisayar Mühendisliği Ana Bilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Prof. Dr. Suat ÖZDEMİR

Bilgisayar Mühendisliği Ana Bilim Dalı, Hacettepe Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ...……

İkinci Danışman: Dr. Mustafa DURSUN

ASELSAN A.Ş.

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ...……

Başkan: Dr. Öğr. Üyesi Yılmaz AR

Bilgisayar Mühendisliği Ana Bilim Dalı, Ankara Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ...……

Üye: Dr. Öğr. Üyesi Uraz YAVANOĞLU

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ...……

Tez Savunma Tarihi: 04/08/2020

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

……..…….

Prof. Dr. Sena YAŞYERLİ

Fen Bilimleri Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Melis BIYIKLI
04/08/2020

RADAR VE ELEKTRONİK HARP PROJELERİNDE "GÖREV PLANLAMA" YAZILIMLARI İÇİN YAZILIM ÜRÜN HATTI MİMARİSİ KAPSAMINDA YENİDEN KULLANILABİLİRLİK YÖNTEMLERİNİN UYGULANMASI

(Yüksek Lisans Tezi)

Melis BIYIKLI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Ağustos 2020

ÖZET

Çalışma kapsamında Elektronik Harp Sistemleri Görev Planlama yazılımının, ürün hattı ile geliştirilmesinde kullanılmak üzere yetenek hattı oluşturulmuştur. Görev Planlama yazılımlarının kullanıcıya sunduğu yetenekler, farklı platformlardaki farklı sistemlerin ihtiyacına göre değişiklik göstermektedir. Bir veya birden çok EH sistemi için ortak kullanılacak Görev Planlama yazılımının hangi yeteneklere sahip olacağı Sistem Tasarım Tanımlama Dokümanı (STTD) dokümanı ve Yazılım Gereksinim Özellikleri (YGÖ) dokümanında yer almaktadır. Bu kapsamda yapılmış olan çalışmada yetenek setlerinin çıkarılması YGÖ dokümanlarından otomatik olarak yapılmıştır. Bu dokümanların çalışmada kullanılabilmesi için Türkçe doğal dil işleme kütüphanesi olan Zemberek kullanılmıştır. Çalışmada kullanılan kelime havuzu, uzman görüşü ve sezgisel yöntemlerle yapılan anket sonucu elde edilen ve yetenekleri tanımlayan kelime ve kelime gruplarından oluşmaktadır. Dokümanlardan ayrıştırılan ifadeler, kelime havuzunda aranmış ve eşleşenler tespit edilmiştir. Havuzdaki kelimelerin hangi yeteneği temsil ettiği önceden belirtilmiş olduğu için eşleşen kelimelerden yetenek tespiti yapılabilmiştir. Böylece STTD dokümanında yer alan isteklerin, Görev Planlama yazılımındaki hangi yeteneğe karşılık geldiği tespit edilebilmiştir. Algoritma sonucu elde edilen yetenekler gerçek projedeki yeteneklerle %80-90 arasında uyumluluk göstermektedir. Yeteneklerin otomatik tespit edilme süreci manuel tespite göre kısalmış ve %50 civarında zamandan kazanç sağlamıştır. YGÖ dokümanları ile başarı oranı arttırılan çalışma sonrasında elde edilen ana ve alt yeteneklerle Görev Planlama yazılımına ait ürün hattı oluşturulmuştur.

Bilim Kodu	:	92408
Anahtar Kelimeler	:	Görev planlama yazılımı, yazılım ürün hattı, yetenek hattı oluşturma, zemberek ile türkçe dil işleme, otomatik yetenek seçimi
Sayfa Adedi	:	75
Danışman	:	Prof.Dr.Suat ÖZDEMİR
2. Danışman	:	Dr.Mustafa DURSUN

APPLICATION OF REUSABILITY METHODS WITHIN THE SCOPE OF SOFTWARE PRODUCT LINE ARCHITECTURE FOR "TASK PLANNING" SOFTWARE IN RADAR AND ELECTRONIC WARFARE PROJECTS

(M. Sc. Thesis)

Melis BIYIKLI

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

August 2020

ABSTRACT

Within the scope of this study, a talent line was created within the scope of the software product line to be used in the development processes of the Mission Planning projects, one of the Electronic Warfare Software, in the Defense Industry Institution. The capabilities offered by Task Planning projects to the user vary according to the needs of different systems on different platforms. The capabilities of the Task Planning project to be used in a system are included in the System Design Described Document (STTD) document and the Software Requirement Features document. In the study conducted within the scope of this study, the extraction of talent sets was made automatically from YGO documents. Zemberek is Turkish natural language processing library. It was used to use these documents in this study. Word pool is the word and word groups that are created as a result of a survey conducted by expert opinion and intuitive methods and define skills. The statements separated from the documents were searched in the word pool and the matching ones were identified. Ability detection can be made from matching words, because the words in the pool represent what skill it has been previously specified. Thus, it was possible to determine which requests in the system document correspond to the capability in the Task Planning project.

Science Code	:	92408
Key Words	:	Mission planning software, software product line, creating talent line, turkish language processing with zemberek, automatic talent selection
Page Number	:	75
Supervisor	:	Prof.Dr.Suat ÖZDEMİR
Co Supervisor	:	Dr.Mustafa DURSUN

TEŞEKKÜR

Çalışmam boyunca bilgi ve deneyimlerinden yararlandığım, ahlaki değerleri ile de örnek aldığım, birlikte çalışmaktan onur duyduğum ve bu süreçte göstermiş olduğu hoşgörü ve sabırdan dolayı değerli hocam Prof. Dr. Suat ÖZDEMİR'e sonsuz teşekkürlerimi sunarım.

Yüksek lisans öğrenim ve tez sürecim boyunca her türlü imkanı sunan ve destek veren ASELSAN Anonim Şirketine, Yazılım Ürün Hattı konusunda ve veri setlerini temin etmekte bana yardımcı olan, konuyu tüm ayrıntılarıyla bana aktaran, her aşamasında yardımcı olan ekip liderim Ozan ÇETİN'e, ikinci tez danışmanım Mustafa DURSUN'a, müdürüm Hasan Hamdi KONYA'ya ve ekip arkadaşlarıma, bilgisi, tecrübeleri ve manevi desteği için dostlarım Kürşat DURAK, Ezgi CANKURTARAN, Selin KARAGÖZ'e ve biricik aileme teşekkürlerimi sunuyorum.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılmış simgeler ve kısaltmalar, açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	**Açıklamalar**
dk	Dakika
sn	Saniye

Kısaltmalar	**Açıklamalar**
EMI	Existence of Meta Information
GPY	Görev Planlama Yazılımı
OSGI	Open Services Gateway Initiative Framework
RCC	Rate of Component Customizability
RCO	Rate of Component
SCCr	Self Completeness of Component's Return Value
SCCp	Self Completeness of Component's Parameter
SO	Standart Output
STTD	Sistem Tasarım Tanımlama Dokümanı
YGÖ	Yazılım Gereksinim Özellikleri

1. GİRİŞ

Teknolojinin gelişimiyle birlikte özellikle son yıllarda artış gösteren sosyal ağ, e-ticaret gibi uygulamalar, büyük miktarda veri toplanmasını sağlamaktadır. Giderek büyüyen veri dünyasında anlamlı verilerle anlamsızlar arasında ayrım yapıp faydalı bilgi elde etme ihtiyacı doğmuştur. Günümüzde pek çok teknik, verilerden anlam çıkarmak ve çeşitli ihtiyaçları karşılamak için kullanılmaktadır. Sağlık, ticaret, borsa, spor gibi birçok alanda eldeki veriyi doğru şekilde kullanıp bilgi elde etme çalışmaları sürmektedir. Verilerden, metinlerden bilgi elde etme işlemleri yazılım mühendisliğinde de kullanılmaktadır.

Geliştirilen yazılımların kalitesini arttırabilmek ve yazılım geliştirme sürecini kısaltabilmek için yazılım ürün hatları oluşturulmaktadır. Yeniden kullanım teknikleriyle yazılımdan ayrılabilen bileşenler başka ürünlerde tekrar kullanılmak üzere ürün hattına yerleştirilmektedir. Bu bileşenlerin tespiti ve sonrasında tekrar kullanımı ile yazılım geliştirme süreci büyük ölçüde kısalmaktadır. Ayrıca önceden test edilmiş bileşenler hata oluşma riskini azaltmaktadır. Bu süreçte yeniden kullanılabilecek bileşenlerin tespiti önemli rol oynamaktadır.

Çalışma kapsamında, elektronik harp sistemleri geliştirme ve entegrasyon işi yapan yazılım müdürlüğünde Görev Planlama Kullanıcı Arayüzü ekibi bünyesindeki yazılımların geliştirilme sürecini kısaltmak için yetenek hattı oluşturulmuştur. Görev Planlama yazılımının kullanıcıya sunduğu yetenekler, farklı platformlardaki farklı sistemlerin ihtiyacına göre değişiklik göstermektedir. Bir sistemde kullanılacak Görev Planlama yazılımının hangi yeteneklere sahip olacağı Yazılım Gereksinim Özellikleri (YGÖ) dokümanında yer almaktadır. Bu kapsamda yetenek setlerinin çıkarılması YGÖ dokümanlarından otomatik olarak yapılmıştır.

Yetenek setlerinin çıkarılması, projeye özgü olanların veya ortak olanların tespiti, YGÖ dokümanından bakılarak geliştirici tarafından manuel yapılmaktadır. Bu yöntem hatalara açık ve zaman alan bir işlemdir. Çalışmada YGÖ dokümanlarından yeteneklerin otomatik tespit edilmesi ve elde edilen yeteneklerin sonrasında yazılım ürün hattına yerleştirilmesi sağlanmaktadır. Çalışmada yeniden kullanılabilecek nitelikte olan bileşenler bilir kişilerce kontrol edilerek tespit edilmiştir. Bu süreçte Hedef-Soru-Metrik yaklaşımı kullanılmıştır. Bir Görev Planlama yazılımında yer alabilecek tüm yetenekler incelenmiştir.

Sonrasında yetenek setleri yazılım geliştirici ekip ile birlikte oluşturulmuştur. Yetenek setleri oluşturduktan sonra her bir ana yeteneği temsil edecek alt yetenekler belirlenmiştir. Bu ana ve alt yetenekler belirlendikten sonra "Uzman Görüşü" kullanılarak doğruluğu kontrol ettirilmiştir. Sonrasında YGÖ dokümanı incelenerek yazılımda yer alması istenen yetenekler tespit edilmiştir. Bu yeteneklerin tespiti için pek çok metot ve kütüphaneden yararlanılmıştır.

YGÖ dokümanları Türkçe dili kullanılarak hazırlandıkları için dokümanlardan kelime çıkarımı için Türkçe doğal dil işleme kütüphanesi olan "Zemberek" kullanılmıştır. Dokümanlardan ayrıştırılan ifadelerle öncesinde N-Gram yöntemi ile çeşitli kombinasyonlar oluşturulmuştur. Sonrasında ise bu ifadeler, kelime havuzunda aranmış ve eşleşenler tespit edilmiştir. Havuzdaki kelimelerin hangi yeteneği temsil ettiği önceden belirtilmiş olduğu için eşleşen kelimelerden yetenek tespiti yapılabilmiştir. Böylece gereksinim dokümanında yer alan isteklerin, Görev Planlama yazılımındaki hangi yeteneğe karşılık geldiği tespit edilebilmiştir. Tespit edilen ana yeteneklerin alt yeteneklerle bağlantıları incelenerek ürün hattı oluşturulmuştur. Doğru yetenek tespiti ve büyük bir ürün hattı için eğitim seti sayısı arttırılarak farklı projelere dair farkı YGÖ dokümanları üzerinde işlemler devam ettirilmiştir.

Tez kapsamında yeniden kullanımı sağlayabilmek için çıkarılan yeteneklerden bir ürün hattı oluşturulmuştur. Bu ürün hattının oluşturulmasında algoritma sonrası elde edilen yetenek bulguları kullanılmıştır. Oluşturulan ürün hattında farklı derinlikte yetenekler yer almaktadır. Başlangıç olarak üç farklı projenin YGÖ dokümanları seçilmiş ve bu dokümanlar eğitim verisi olarak kullanılmıştır. Ürün hattının değiştirilip düzenlenebilir yapısıyla diğer YGÖ dokümanları ve projelerle birlikte gelişmeye devam etmektedir. Ürün hattındaki özellikler ile proje- yetenek eşleşmesi, akran gözden geçirme (İng. peer review) kullanılarak doğrulanmıştır. Böylece eksiksiz ve hatasız bir şekilde özellik ağacının çıkarılması sağlanmıştır.

Çalışma Türkçe metinlerden otomatik yetenek çıkarılmasını sağladığı için hem Türk Silahlı Kuvvetlerinin bir şirketindeki elektronik harp görev yazılımların ürün hattı oluşturabilmesi ve proje geliştirme sürecine olumlu katkı sağlayabilmesi açısından tek örnek olacaktır. Literatüre de Türkçe dili üzerinde yapılan çalışmalar ve örnek veri seti oluşturma çalışmaları açısından ilham kaynağı olacaktır.

Tez çalışmasının geriye kalan kısmı şu şekilde oluşturulmuştur: 2. Bölümü'nde literatürde kullanılan yeniden kullanılabilirlik çalışmaları ve Türkçe doğal dil işleme çalışmaları açıklanmaktadır. Ayrıca, yazılım ürün hattının yeniden kullanılabilirliği hakkındaki literatür çalışmalarına da yer verilmiştir.

3. Bölümü'nde yapılan çalışmaların bağlamı ve çalışmada izlenen yöntem açıklanmaktadır. 4. Bölümde çalışma ve test sonuçları ayrıntılı şekilde anlatılmıştır. 5. Bölümde ise 4. Bölüm 'deki çalışmaların sonuçları değerlendirilmektedir. Bulgular genel olarak değerlendirilmekte, kullanımına yönelik riskler anlatılmakta ve gelecek çalışmalarından bahsedilmektedir.

2. İLGİLİ ÇALIŞMALAR

Bu bölümde yeniden kullanılabilirlik çalışmaları, Türkçe doğal dil işleme çalışmaları ve yazılım ürün hattının yeniden kullanılabilirliği ile ilgili çalışmalara yer verilmiştir.

Literatüre bakıldığında Sandhu ve Singh çalışmalarında yeniden kullanılabilirlikle ilgili metrikler öne sürmüşlerdir. Yeniden kullanılabilirliği 6 farklı metrikle (perfect, high, medium, low, very low ve nil) değerlendiren sistem tasarlamışlardır. Değerlendirmenin başarımını arttırma için iki yüzden fazla iterasyon uygulamışlardır. Her altı sınıf için de ayrı ayrı kırk sekiz parametre atamışlar ve değişkenlerin girdi parametrelerine atanmalarını sağlamışlardır. Yaptıkları çalışmalarla sadece yeniden kullanılabilir bileşenlerin kalitesini ölçmeyip aynı zamanda nörobulanık sistemler için bulanık kuralların nasıl seçilebileceğine de ışık tutmuşlardır [1].

Torkamani, yazılım parçalarının değerlerini kullanarak yeniden kullanılabilir bileşenlerin kalitesini ve yeniden kullanılabilirlik oranını ölçebilen metrik geliştirmiştir. Bileşenlerin yeniden kullanılmasından dolayı elde edilen maliyet, zaman karı değerlerini ortaya koyarak yeni bir ürünün oluşturulmasındaki etkilerini ortaya koymuştur [2].

Ürün hattı mimari dili özel ürün hattı mimarilerini tanımlamada ve hattın kalitesini ölçmede kullanılmaktadır. Zhang ve arkadaşları doğrudan yeniden kullanılabilirlik kalitesini ölçen metriklerden ziyade yeniden kullanılabilirlik, karmaşıklık, benzerlik ve farklılık gibi metriklerin ürün hattı kalitesine nasıl etki ettiklerini incelemişlerdir. Bu ölçümleri yaparken ADL olarak isimlendirilen bu dili kullanmışlardır [3].

Korra ve arkadaşları, yeniden kullanılabilir bileşenleri yaratabilmek ve yönetebilmek amaçlı yazılım mühendisliği metot ve stratejileri sunmuşlardır. Çalışmada ayrıca yazılım bileşenlerinin farklı veri modellerine uyarlanabilmesi ve taşınabilmesi için özel uygulamalar sunmuşlardır. Gelişen sistemler, değişken istekler ve deneyim azlıkları yeniden kullanımda hatalara yol açmakta ve önünü kapayabilmektedir. Tutarlı, hatadan arındırılmış, bağımsız ve güvenilir parçalar yeniden kullanıma imkân sağlamaktadır. Yapılan çalışmada yeniden kullanılabilir ögelerin oluşturulmasında standartlaştırma, sertifikalandırma, genelleştirme ve otomatikleştirme adımlarının izlenmesi gerektiğine değinilmiştir [4]. Basili ve Çaldiera yeni metotlar önermiştir.

Bu parçaları içermek, yeniden kullanılabilirlik için gereken özellikler arasında sayılmıştır. Yazılımlarda yeniden kullanım oranını arttırabilmek için farklı süreçler sunmuşlardır [5]:

- Mevcut potansiyelin ölçülerek, yeniden kullanım uygunluğunun tespit edilmesi
- Kullanılacak bileşenlerin standartlarının ölçümü
- Fayda analizinin yapılması
- Mevcut metriklerin değerlendirilmesi
- Örnek olması için pilot projelerin seçimi

Öne sürdükleri kriterlerle mevcut durumun analizini yaparak, hedefler koymayı ve örnek projelerle yeniden kullanımı pekiştirmeyi vurgulamışlardır. İyi bir planlama ve yönetimle başarılı yeniden kullanım yapılabileceğini öne sürmüşlerdir. Aynı şekilde fayda-maliyet analizinin de yeniden kullanılır bileşenlerin başarımı hakkında bilgi verdiğini ve buradan çıkan bilgiyle yatırım yapılıp yapılamayacağını belirtmişlerdir [5].

Younoussi ve Roudies yeniden kullanılabilirliği etkileyen özellikleri ve bu konuda yapılan farklı yaklaşımları sistematik olarak literatürde incelemişler ve ortaya birleşik bir çalışma çıkarmışlardır. Yeniden kullanımı etkileyen faktörler olarak "anlaşılabilir, taşınabilir, bakıma uygun, adapte edilebilir, esnek, sabit, dokümante edilebilir ve bağımsız olunabilir" niteliklerini vurgulamışlardır. Yazılım bileşeninin anlaşılabilir olması, yorum satırları ve dokümanlarla desteklenmesi yeniden kullanım için kolaylık sağlamaktadır. Taşınabilir bir birleşen rahatlıkla ürün hattına yerleştirilebilir ve bir yazılımdan diğerine çok az değişikliklerle taşınabilir. Esneklik farklı konfigürasyonlarla farklı kullanımlara uyum sağlayabilmektir. Sabitlik olgunlaşmışlığın ölçüsüdür. Yazılım bileşeni farklı kullanımlarda hatalara ve büyük değişikliklere yol açmamalıdır. Kendi görevini başarıyla yerine getiren yazılım parçası bağımsız olarak nitelendirilebilirken, anlaşılırlığını ve yeniden kullanımını arttırma amaçlı iyi bir dokümantasyona sahip olması da önemli olmaktadır [6].

Babu ve Srivatsa'nın yaptığı çalışma bir yazılımdan maksimum ne kadar verimin nasıl alınacağını vurgulamaktadır. Yeniden kullanımın ne kadar olabileceği, mevcut kodun ne kadarının kullanılabileceğinin ölçümünü ve ortaya çıkan maliyet anlatmaktadır [7]. Torkamani ve arkadaşlarının çalışmasında yazılım ürün hattı yaklaşımı vurgulanmıştır. Bileşenlerin yeniden kullanılabilirlikleri değerlendirilmiştir. İşlevsellik, uyumluluk ve modülerlik yeniden kullanılabilir parçalar için vazgeçilmezdir.

Sonrasında ürün hattının başarısının doğrudan arttırıldığına dikkat çekilmiştir. Yeniden kullanılabilir bileşenlerin kalitesini ölçebilmek için pek çok metrik sunmuşlardır. Her bir bileşene özgü kalite özellikleri belirlenmiş ve metrikler tanımlanmıştır. Metrikleri güvenilirlik, kullanılabilirlik, taşınabilirlik, bakım yapılabilirlik, işlevsellik ve verimlilik olarak nitelendirmişlerdir. Bu bağlamda ISO 9126 standardı çerçeve alınarak yeni bir yaklaşım geliştirilmiştir. Buna bağlı olarak ürün hattını değerlendirme amaçlı kılavuz sunmuşlardır [8].

Washizaki çalışmasında herhangi bir kod bilgisi olmadan geliştirici bakış açısı ile yeniden kullanılabilirliği ölçmeyi hedeflemiştir. Taşınabilirlik, adapte olabilirlik ve anlaşılabilirlik ile yeniden kullanılabilirliğe dikkat çekmiştir. Bu faktörleri kendi içlerinde "SCCr, EMI, RCC, RCO ve SCCp" olmak üzere beş alt faktörle ve bunların kombinasyonlarıyla temsil etmiştir [9].

Atkinson ve Muthig'in çalışmasında gereksinimlerin birbiriyle çatışması, bileşenlerin bağımlı olması ve kaliteli kod ortaya çıkaramamanın yeniden kullanımı azalttığını ve ürün hattının başarımını düşürdüğünü vurgulamıştır. Ürün hattını geliştirmek için bileşen tabanlı bir yaklaşım kullanmıştır. Ürün hattı mühendisliği ile Model-Driven Architecture (MDA) yaklaşımını birleştirerek yeniden kullanımda maksimum verim almayı amaçlamıştır [10].

Davidov ve arkadaşları, twitter mesajlarında duygu analizi yapmışlardır. Metin işleme teknikleriyle birlikte makine öğrenmesinde yer alan K-NN sınıflandırma tekniğini kullanarak bir test verisinin hangi duyguya daha yakın olacağını sınıflandırabilmişlerdir. Kelimeleri sık geçme frekans yöntemiyle inceleyerek eğitim setini algoritmayı eğitme amaçlı kullanmışlardır. Çalışmanın girdilerini genişleterek emojileri ve hashtagleri de duygu analizine dahil etmişlerdir. Önceden oluşturduklarını veritabanı ile hangi ifadenin, hangi sınıfa benzerlik gösterdiğini k-nn yöntemi ile belirlemişlerdir [11].

Taboada ve arkadaşları yorumlardan duygu çıkarımı için çalışma yapmışlardır. Bunun için kendi sözlüklerini oluşturmuşlardır. Duygu terimleri sözlüğünü oluşturabilmek için "Mechanical Turk" isimli servisi kullanmışlardır. SentiWordNet, Google PMI, Maryland gibi diğer sözlüklerle kıyaslayarak kendi sözlüklerinin daha kapsamlı, daha güvenilir ve başarılı olduğunu test sonuçlarıyla göz önüne sermişlerdir [12]. SO-CAL (the Semantic Orientation CALculator) tekniğiyle fill ve sıfat gibi cümlenin ögelerini ayırmışlardır.

Sonrasında her birinin ifade ettiği anlam için SO değerini hesaplamışlardır. Hesaba göre Çizelge 2.1'deki gibi kelime ve değer tabloları oluşturmuşlardır. Hesaplamayı "sleazy" kelimesi SO değeri: −3 ise "somewhat sleazy" kelimesi SO değeri: −3 × (100% − 30%) = −2.1 ve "excellent" kelimesi SO değeri: 5 ise "most excellent" kelimesi SO değeri: 5 × (100% + 100%) = 10. Olacak şeklinde yapmışlardır [12].

Çizelge 2.1. Bazı vurgulayıcılar için yüzdeler [12]

Intensifier Modifier (%)	Intensifier Modifier (%)
slightly −50	slightly −50
somewhat −30	somewhat −30
pretty −10	pretty −10
really +15	really +15
very +25	very +25
extraordinarily +50	extraordinarily +50
(the) most +100	(the) most +100

Sonrasında yorumlardan çıkardıkları duyguları zamir, sıfat, yüklem oluşlarına göre farklı kategorilerde ve farklı derecelerle nitelendirmişlerdir. Olumlu ve olumsuzluk derecelerine göre -5 ile 5 arasında değerler atayarak sınıflandırmışlardır. Bu çalışma İngilizce terimler üzerinde gerçekleştirilmiştir. "funny" kelimesi +2 değere sahipken "act funny" olumsuz anlam taşıdığı için -1 ile derecelendirilmiştir [12].

Khan F. H. ve arkadaşları online sinema yorumları üzerinden duygu analizi çalışması yapmışlardır. Bunun için hazır veritabanı olan SentiWordNet sözlüğünü kullanmışlardır. Duygu analizi için java dilinde eSAP (Enhanced Sentiment Analysis and Polarity) frameworkünü kullanmışlardır. İngilizce metinleri ön işlemeden geçirmişlerdir. Boşlukları ve noktalama işaretlerini temizledikten sonra her bir kelimeyi sözlük yardımıyla etiketlemişlerdir. SentiWordNet sözlüğünü geliştirerek kendi kütüphanelerini oluşturmuşlardır. Test için SVM (Support Vector Machine) tekniğini kullanmışlardır [13]. Eroğlu 2009'da yaptığı çalışmada Türkçe sinema yorumları üzerinde duygu analizini gerçekleştirmiştir. Literatürü incelediğinde yabancı sinema yorumlarının daha detaylı ancak Türkçe yorumların genel hatlarla ve bir veya iki cümleden oluştuğunu gözlemlemiştir. Sinemalar.com ve beyazperde.com sitelerindeki yorumları otomatik alan ve web sitesi formatındaki %d, %s ifadelerini temizleyen algoritma geliştirmişlerdir [14]. Sonrasında Türkçe doğal dil işleme kütüphanesi olan "Zemberek"i kullanmıştır. Cümleleri kelime yapılarına göre gruplandırdıktan sonra ifadelerden duyguları çıkarmıştır.

Test verileri için n-gram metodu ve SVM tekniğinin de yardımıyla metinleri sınıflandırmıştır. Çalışma sonunda sınıflandırma doğruluğu için F-skor ve F1 Ölçüm tekniklerini kullanmıştır [14].

Kaya ve arkadaşları, Türkçe haber sitelerindeki politika haberleri üzerinde duygu analizi çalışması gerçekleştirmişlerdir. Türkçe metinlerdeki kelimeleri ve türlerini ayırt edebilmek için Zemberek kütüphanesinden faydalanmışlardır. Sıfat, zarf ve fiil kökenli sözcükleri ayrıştırdıktan sonra olumlu ve olumsuz duyguları ifade eden sözcük veri seti oluşturmuşlardır. Çalışma için Naïve Bayes, Maximum Entropy, SVM ve karakter tabanlı N-Gram olmak üzere dört farklı makine öğrenmesi tekniği kullanmışlardır. Sınıflandırma test sonuçlarında %81.5 ile Maximum Entropy ve N-Gram yöntemlerinin Naïve Bayes ve SVM'den daha başarılı sonuç verdiğini gözlemlemişlerdir [15].

Şimşek ve Özdemir, 2012'de twitter verileri üzerinde çalışma yapmışlardır. Çalışmada mesajlardan duygu analizi yaparak çıkan sonuçları market stoklarıyla ilişkilendirmişlerdir. Mesajları veri madenciliği teknikleriyle elde ettikten sonra kendi oluşturdukları 113 sözcüklü veri setindeki kelimelerle kıyaslamışlardır. Bu veri setini mutlu ve mutsuz şeklinde iki sınıf olarak oluşturmuşlardır. Metinler içinde bu kelimeleri aratarak duygu çıkarımı yapmışlardır. Bu süreçte twitter üzerindeki özel mesajları seçmişlerdir. Sonuçlara göre stokun az olduğu zaman atılan mesajlar mutsuz ifadeler içerirken, stokun fazla olduğu zamanlar atılan mesajlar mutlu ifadeler içermektedir. Stoklara bağlı olarak yıl sonu zamanlarındaki mesajların mutlu edici ve pozitif ifadeler olduğunu gözlemlemişlerdir [16].

Aytekin, yaptığı çalışmada Türkçe blog sitelerdeki yorumlar üzerinde duygu analizi gerçekleştirmiştir. Google Blog Search Engine yardımıyla metinlerden anahtar kelimelerin çıkarılmasını sağlamıştır. İngilizce duygu ifadeleri veri tabanlarından çevirdiği kelimelerden oluşan sözcükleri metin içinde aratmıştır. Farklı duygu ifadeleri için farklı renklendirmeler, RGB kodları ve 108 farklı hexadecimal kod kullanmıştır. Sözcük tabanlı yöntem ile makine öğrenmesi yöntemlerini birleştirerek sözcüklerin hangi duygu ifadelerine benzediğini ölçebilmiştir. Çalışması sonrasında 350 pozitif yorumda %72 ve 350 negatif yorumda %73 doğruluk oranı ölçmüştür [17].

Sevindi, literatürde duygu analizi için sıkça kullanılan sözcük tabanlı yöntemle makine öğrenmesi yöntemlerini kıyaslamıştır.

Çalışmasında 1057 pozitif ve 978 negatif sinema yorumunun elle etiketlendiği veri seti kullanılmıştır. Sözcükleri ön işleme için Turkish Deasciifier programının kütüphanesini kullanarak UTF-8 formatında elde edebilmiştir. Kelimeleri daha ayrıntılı inceleyebilmek için Zemberek kütüphanesinden faydalanmıştır. Duygu veri seti olarak WordNet kütüphanesini kullanılmıştır.

Duygu sınıflandırma işlemlerinde SVM yönteminin K-NN algoritmasına göre daha başarılı olduğunu gözlemlemiştir. Sözcük tabanlı yöntem yaklaşımında terim ağırlıklandırma yöntemini kullanarak kelimeleri ve geçme frekanslarını incelemiştir. Tf*idf skoru en yüksek olan kelimelere odaklanmıştır. Görüş çıkartma işlemlerinde dilsel örnekleme yerine N_Gram yöntemleri (unigram, bigram ve 3-gram) tercih etmiştir. Görüş sözlüğü yaklaşımıyla Naive Bayes ve Maximum Entropy sınıflandırıcıları karşılaştırılmıştır. Sonrasında ise SVM ve C5 karar ağacı algoritmalarıyla sözlük tabanlı yaklaşım kıyaslanmıştır. Sonuç olarak TF-TDF ağırlıklandırma yönteminin diğer yöntemlerden daha başarılı sonuç verdiğini gözlemlemiştir [18].

Li ve arkadaşlarının yaptığı çalışmada SRS (Software Requirements Specifications) dokümanlarından ön gereksinimlerin (amaç, kısıtlamalar ve fonksiyonlar gibi) otomatik çıkarımı yapılmıştır. Çalışmada doğal dil işleme teknikleri kullanılmıştır. Tanımlanan desen ile eşleşen girdi parametreleri bulunarak yetenek tespiti yapılmıştır. Çalışmada "Gaussian Elimination", "Monte Carlo" ve "Finite Element" metotlarıyla birlikte part-of-speech (POS) etiketleyicisi kullanılarak kelimelerin eş anlamlıları ve olası varyasyonları çıkarılmıştır. Veri setine göre Matlab Topic Modeling aracı yardımıyla dokümandaki "gereksinim" niteliğindeki cümlelerden anahtar kelimeleri çıkarabilmişlerdir. Radyan değeri yüksek olan kelimelerin yetenek olarak mevcut olduğu sonucunu çıkarmışlardır. Testler sonrasında %78-97 arasında doğruluk tespit edilmiştir [19].

Mu ve arkadaşları SRS dokümanlarından fonksiyonel gereksinimlerin çıkarılması için çalışma yapmışlardır. Bunun için SRS dokümanı yapısını inceleyerek 10 farklı anlamsal durum belirlemiş ve kurallar oluşturmuşlardır. Doğal dil işleme tekniklerini kullanarak EFRF (Extended Functional Requirements Framework) frameworkü inşa etmişlerdir. Sonuçları tüm ürün hattın kullanmışlardır. Test sonuçlarından uygulanan yaklaşımın doğruluğunu, ölçeklenebilirliğini ve genişletilebilir olduğunu gözlemlemişlerdir [20]. Slankas ve Williams, fonksiyonel olmayan gereksinimleri dokümanlarda aratmışlardır.

Kelime vektörleri oluşturularak K-NN ve Naïve Bayes makine öğrenmesi teknikleri kullanılmıştır. Dokümandan fonksiyonel olmayan gereksinimleri yakalayabilmek için literatürde en sık geçen 14 ifade algoritmada kullanılmıştır. Metinlerden kelimelerin ve sıfat, fiil, zamir gibi türlerinin elde edilmesi için Stanford Natural Language Parser (NLP) aracı kullanılmıştır. Dokümanda yer alan mevcut kelimeler, literatürde araştırıp buldukları ve hali hazırda kullanılan kelime gruplarıyla karşılaştırılarak benzerliği ölçülmüştür. Çalışma sonunda gereksinimlerin tespitinde K-NN algoritmasının F1 skor değerini 0.54 olarak ve Naïve Bayes algoritmasının F1 skor değerini 0.32 olarak ölçmüşlerdir [21].

Cleland-Huang ve arkadaşları çeşitli dokümanlardan fonksiyonel olmayan gereksinimlerin çıkarımı için çalışma yürütmüşlerdir. Uzmanlarca oluşturulan SIG (Softgoal İnterdependency Graphs) kataloğunda Şekil 2.1'deki gibi her bir fonksiyonel olmayan gereksinim için anahtar kelimeler tanımlamışlardır [22].

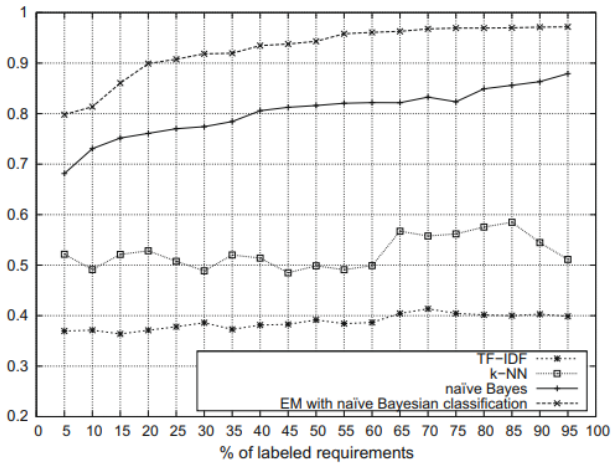
NFR Type	**Keywords extracted from a SIG Catalog [6]**
Security	Confidentiality, integrity, completeness, accuracy, perturbation, virus, access, authorization, rule, validation, audit, biometrics, card, key, password, alarm, encryption, noise.
Performance	space, time, memory, storage, response, throughput, peak, mean, index, compress, uncompress, runtime, perform, execute, dynamic, offset, reduce, fixing, early, late

Şekil 2.1. Fonksiyonel olmayan gereksinimler için anahtar kelimeler [22]

NFR (Non-functional Requirements) metodu ismini verdikleri kendi geliştirdikleri algoritma ile dokümanlarda yer alan gereksinimleri yakalayabilmişlerdir. Metot performansı sonuçlarını incelediklerinde literatürdeki pek çok yarı otomatik sınıflandırma algoritmalarından başarılı olduğunu gözlemlemişlerdir. "Integrated Engineering Toolset" ismi verilen hazır verileri kullanmışlardır. IET veri setiyle fonksiyonel olmayan gereksinimlerin daha iyi tespit edildiği sonucuna varmışlardır [22].

Casamayor ve arkadaşları fonksiyonel olmayan gereksinimlerin tespiti için yeni bir yaklaşım sunmuşlardır. Bu yaklaşımda daha az sayıda kategorize edilmiş gereksinim mevcuttur. Kullanıcılardan gelen geri bildirimlerle zamanla gelişen bir sınıflandırma algoritması geliştirmişlerdir [23].

Algoritmanın temelini Naïve Bayes oluşturmaktadır. Metin ön işlemeden sonra "PROMISE Software Engineering" veri seti kullanılarak eğitilen algoritma, birçok uzmandan gelen geri bildirimleri toparlayarak sık geçen ve doğruluğu yüksek olan yeni gereksinimleri de hesaplamaya katmaktadır [23].



Şekil 2.2. Çeşitli sınıflandırıcıların başarı oranı [23]

Kendi kendine yineleyen yapısıyla sınıflandırma doğruluk başarısını arttırarak Şekil 2.2'de görüldüğü gibi diğer sınıflandırma algoritmalarından daha yüksek başarı elde etmiştir. Yapılan yinelemelerle başarımın arttığını ölçmüşlerdir. 10 yineleme ile %75 doğruluk oranı ölçerken, 30 yinelemeli öğrenme işlemiyle %100 başarı oranı yakalamışlardır [23].

3. UYGULANAN YÖNTEM VE METOTLAR

3.1. Yeniden Kullanım

Yeniden kullanım hayatın pek çok yerinde rol oynayan bir süreçtir. Yazılım geliştirme süreçlerinde üretim hızını ve yazılım kalitesini arttırmak için vazgeçilmez nitelikte olmaktadır. Yazılımlar arasında ortak olan ve farklı olan yeteneklerin tespitinden sonra ortak olanları soyutlayabilmek ve diğer projelerde tekrar kullanabilmek yeniden kullanımın en büyük getirilerinden bir olmaktadır [24].

Ramamoorthy ve Sommerville'e göre yeniden kullanımdaki problem, yeniden kullanılabilir bileşen sayısındaki azlıktır. Etkili yeniden kullanım tasarımın, analizin ve dokümanın da yeniden kullanılmasıdır. Yeniden kullanım ile geliştirme yapabilmek için ön saklama süreci, ön kullanım süreci ve yeniden kullanılabilir bileşenlerin depolanması süreci başarıyla yürütülmelidir [25]. Bu bileşenlerin sayısının artması ve yazılım modüllerinin soyutlanabilir olması ürün hattının başarılı oluşturulabildiğine ve yazılım kalitesinin yüksek olduğuna işaret etmektedir.

3.2. OSGI ve Modülerlik

Tez çalışması kapsamında ürün hattı, GPY projeleri için oluşturulmuştur. Görev Planlama yazılımı java programlama dilinde ve OSGI çerçevesinde yazılmaktadır. OSGI çerçevesinde geliştiriliyor olması modülerliğe katkı sağlamaktadır. Yazılım parçalarının bir bütün olmayıp farklı proje ve ihtiyaçlara göre bağımsız birer parça olmaları önem arz etmektedir. Bu sayede ihtiyaca göre indirgenip yükseltgenebilmektedirler. Çalışma için belirlenecek ana ve alt yetenekler bu projelerin özelliklerine göre tespit edilecektir. Bir projeden soyutlanabilir, bağımsız olarak geliştirilebilir ve farklı bir projeye entegre edilebilir yapılar belirlendikten sonra YGÖ dokümanı içinden tespit edilecektir [26].

3.3. Zemberek Kütüphanesi

Bu bölümde ürün hattına yerleştirmek için YGÖ dokümanlarından yeteneklerin nasıl elde edildiğine dair bilgi verilmektedir. YGÖ dokümanları Türkçe dili kullanılarak oluşturulmuş belgeler olduğu için yeteneklerin çıkarılmasında Zemberek doğal dil işleme kütüphanesi kullanılmıştır. Literatüre bakıldığında Wang [27], SRS dokümanlarından anlamlı bilgi elde edebilmek için anlamsal rol etiketleme ile alan bilgisi modelleme tekniklerini birleştirerek yeni bir yöntem ortaya atmıştır.

Bu modül sayesinde morfolojik analiz ve kelime türetme işlemleri kolayca yapılabilmektedir. Tez çalışmasında yetenek çıkarımı Zemberek kütüphanesi ile yapıldıktan sonra doğru kelimelerin elde edilebilmesi için başka düzenleme işlemleri de yapılmıştır.

3.4. Ürün Hattı Oluşturma

Bu bölümde yeniden kullanılabilirliği sağlayabilmek ve başka çalışmalarda uygulanabilir olmasını sağlamak için tez çalışması sonunda çıktı olarak elde edilen ürün hattının nasıl oluşturulduğuna dair bilgi verilmektedir. Literatürde SRS dokümanından özellikleri çıkarabilmek ve ürün hattı oluşturabilmek için Hamza ve Walker [28] doğal dil işleme tekniklerinden FFRE yöntemini kullanmıştır. SRS dokümanından özellikleri ve özellikler arası bağlantıları çıkarmışlardır. Ürün hattı yaklaşımı 1990'lardan itibaren hayata geçirilmiştir. Olgunlaşan ve testi yapılmış yeteneklerle daha hızlı ve kaliteli yazılım geliştirmeyi esas alan bir yöntemdir. Temeli yeniden kullanımı büyük ölçüde desteklemektedir [29].

3.5. N-Gram Algoritması

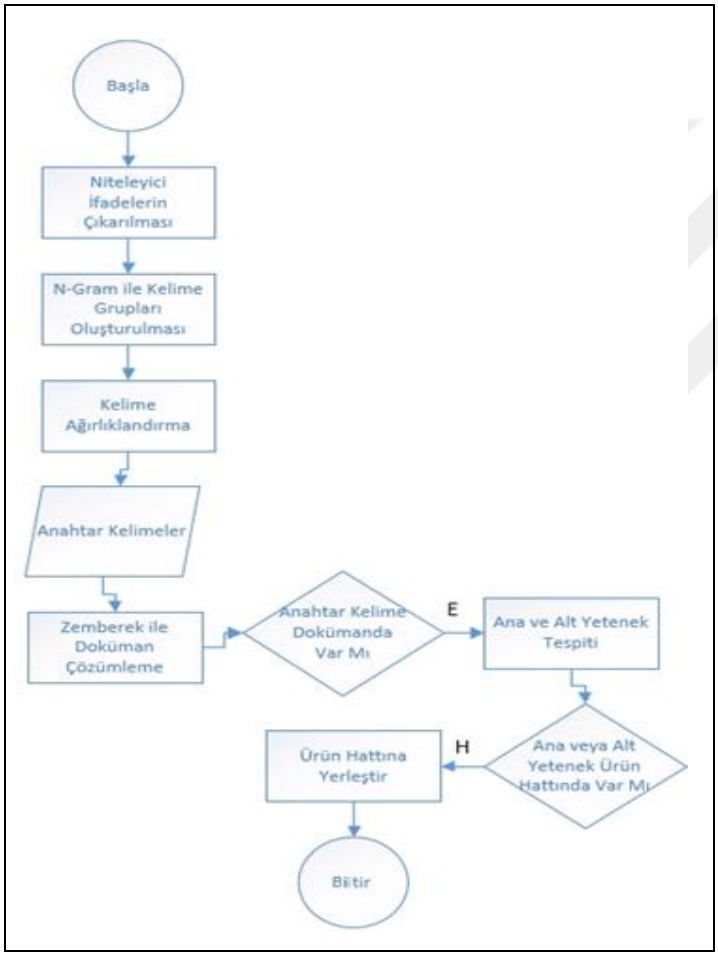
Bir karakter grubunun daha alt karakterlere bölünerek ifade edilmesi ile oluşurlar. Kelime tamlamaları tekli, ikili veya üçlü kelimelerle ifade edilebilirken bir kelime ise hecelerine ayrılarak farklı n-gram isimlendirmeleriyle temsil edilebilir. Bir kümedeki elemanın bulunma durumunu ve sıklığını ölçmede, arama algoritmalarında ve sınıflandırma problemlerinde sıkça kullanılmaktadır. N değeri karakterlerin kaçar kaçar bölüneceğini ve gram değeri de ilgili karakterin veri setinde geçme sıklığını ifade etmektedir [30].

3.6. Metin Madenciliği

Metin madenciliğinin temel konusu farklı türdeki metinlerin sınıflandırılması ve metinlerden anlamsal ifadelerin çıkarılmasına yöneliktir. Bu alanda yapılan çalışmaların çoğunda doğal dil işleme teknikleri kullanılmaktadır. En önemli nokta eğitim veri setinin büyüklüğüdür. Güvenlik, sosyal medya, reklamcılık gibi pek çok alanda kullanımı mevcuttur. Tez kapsamındaki çalışmada metinlerden anlamsal bilgi çıkarmak ve amaç doğrultusunda işleyebilmek adına çalışma yürütülmüştür [31].

4. YETENEK TESPİTİ VE ÜRÜN HATTI TASARIMI

Bu bölüm genelinde yapılan çalışma ve çalışma sonuçları hakkında bilgi verilmiştir. İlk olarak Görev Planlama yazılımında kullanılan yetenek setlerinin ve yetenekleri niteleyici ifadelerin nasıl oluşturulduğu ve hazırlık çalışmaları anlatılmıştır. Yeteneklerin belirlendikten sonra YGÖ dokümanlarındaki ifadelerden yeteneklerin nasıl elde edildiği açıklanmıştır. Dokümanlardan çıkarılan yeteneklerin doğruluğunun nasıl ölçüldüğü, test çalışmaları ve sonuçları ayrıntılı olarak anlatılmıştır.



Şekil 4.1. Uygulama akış şeması

Çalışma esnasında Şekil 4.1'de belirtildiği gibi öncelikle yetenekleri niteleyecek ifadeler oluşturulmuştur. N-Gram yöntemi ile farklı alt kelime grupları oluşturulduktan sonra anahtar kelimeler belirlenmiştir. Zemberek kütüphanesi ile dokümanın çözümlenmesinden sonra anahtar kelime eşleşmesine bakılmıştır. Ana ve alt yeteneklerin tespit edilmesinden sonra yeteneğin ürün hattında önceden olup olmadığına bakılmıştır. Ürün hattında olmayan yetenekler hiyerarşik olarak hatta yerleştirilmiştir.

4.1. Hedef-Soru-Metrik Yaklaşımı ile Kapsamın Belirlenmesi

Tez çalışmasında kullanılacak veri setini ve metrikleri belirleyebilmek için Hedef-Soru-Metrik (İng. Goal-Question-Metric) yaklaşımı uygulanmıştır. Bu yaklaşımda amaçlı ölçüm varsayımı mevcuttur. Amaçlı ölçüm varsayımında projeye dair hedefler tanımlanmaktadır. Elde edilmek istenen bilgi tespit edilir. Problemin çeşidine göre bilgiler sayısallaştırılır. Üç farklı seviye mevcuttur.

i. Kavramsal Seviye (Amaç): Farklı bakış açıları ve çeşitli nedenler için hedef tanımlanan seviyedir. Ölçülen nesne; dokümanlar, kaynak kodlar, geliştirme süreci, test süreci, çalışan, yazılım gibi değerler olabilir [10].
ii. Operasyonel Seviye (Soru): Sorular ile ölçüm nesnesi karakterize edilmeye çalışılır. Hedefleri değerlendirme ve netleştirme için kullanılan sorular setidir [10].
iii. Nicel Seviye (Metrik): Sorularla ilişkili olan veri setidir. Metrikler objektif ya da sübjektif olabilir. Objektif metrikler nesnenin kendisine bağlıdır ve ölçülebilen ispatlanabilen değerlerdir (dokümanın sürüm numarası, program boyutu vb.). Sübjektif metrikler ise bakış açısına bağlı olan değerlerdir (Örneğin kullanıcının memnuniyet derecesi) [10].

Hedef kısmında ölçümün konusu ve amacı belirlenir. Hedef içinde objektif veya sübjektif metrikler yer alabilmektedir. Yöntem ise yazılım kalitesini arttırmada kullanılmaktadır. Çalışmanın girdilerinin ve çıktılarının önceden belirlenmesi için sorular oluşturulmuş ve çalışmanın kapsamını belirleyebilmek için hedefler konmuştur. Hedefe ilerleme aşamasında pek çok metrikten faydalanılmıştır. Bu metriklere YGÖ dokümanları, yazılım ara yüzleri, geliştirici yorumları örnek verilebilir.

STTD dokümanları tüm projelerin yeteneklerini kapsamaktadır. YGÖ dokümanları ise projelere göre özelleştirilmiş olduğu için daha hedef odaklı bir kapsama sahiptir. Bu yüzden çalışma için gerekli olan girdilerin YGÖ dokümanları olduğuna karar verilmiştir. YGÖ dokümanlarından elde edilecek bazı kelimelerin gerçekte bir yetenek ifade edebilmesi ve diğer kelimelerden ayırt edilebilmesi için öncelikle bir kelime havuzu oluşturulmuştur. Bu havuzun oluşturulmasında sübjektif bir metrik olan geliştirici yorumları kullanılmıştır. Görev Planlama yazılımında yıllarca görev almış, alanında uzman kişiler tarafından yazılımın ana ve alt yetenekleri belirlenmiştir. İncelemeler sonucunda hali hazırda altı ana yetenek ve kırk beş alt yetenek tespit edilmiştir. Girdiler netleştirildikten sonra çalışmada izlenecek yöntem tasarlanmıştır.

YGÖ dokümanının incelenerek belirlenen yetenekleri vurgulayan ifadelerin yakalanmasına ve yakalanan ifadelerin hangi ana ve alt yeteneğe karşılık geldiğinin tespit edilmesine karar verilmiştir. Tespit edilen yetenekler ürün hattına yerleştirilmiştir. Çalışmanın amacı olarak yeni gelen bir projenin YGÖ dokümanının incelenmesi ve Görev Planlama yazılımının hangi yeteneklerini istediğinin otomatik olarak tespiti hedeflenmiştir.

4.2. Niteleyici Kelime Setlerinin Oluşturulması

Görev Planlama yazılımında kullanılan ana yetenekler yazılım geliştirici ekip tarafından detaylı çalışmalar sonucunda çıkarılmıştır. Ana yetenekler belirlendikten sonra her bir ana yeteneği temsil eden alt niteleyici yetenekler belirlenmiştir. Ana ve alt yetenekler belirlendikten sonra bu yetenekleri temsil eden kelime grupları oluşturulmuştur. Doküman içinde geçen hangi kelimenin hangi yeteneği temsil edeceğini belirleyebilmek için kelime-yetenek matrisi oluşturulmuştur. Ancak bazı kelime veya kelime gruplarının yetenekleri temsil etmede diğerlerine oranla daha büyük rol oynadığı kararına varılmıştır. Bu yüzden kelime*yetenek matrisinde yer alan her bir kelime için ağırlıklandırma çalışması yapılmıştır. Her bir kelimenin yeteneği temsil etmedeki etkisi belirlenmiştir. Örnek olarak yetenek ve niteleyicileri Çizelge 4.1 ve Çizelge 4.2'de verilmiştir.

Çizelge 4.1. Niteleyici kelime - senaryo yetenek matrisi

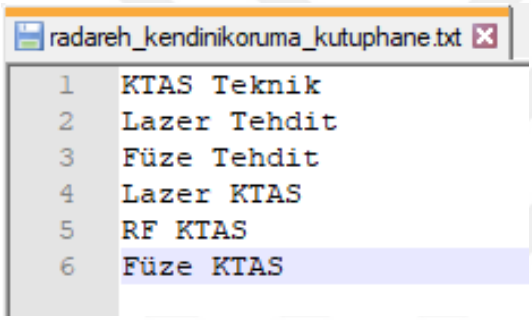
	radariceaktarma/radareh/senaryo
Yetenek 1	5
Yetenek 2	1
Yetenek 3	10
Yetenek 4	10

Çizelge 4.1'den Yetenek 3 ve Yetenek 4 ifadelerinin "Senaryo" ana yeteneğinde bulunan Radar Elektronik Harp grubundan içe aktarma yeteneğini daha yüksek puanla temsil ettiği sonucu çıkarılmaktadır. İlgili ana yeteneği temsil etmede daha ayırt edici rol oynadıkları için ağırlıkları sayısal olarak en yüksek değer verilmiştir. Yetenek 1 başka yeteneklerde de geçebilmektedir ve ilgili yetenek için doğrudan bir ayırt edicilik oluşturmamaktadır. Yetenek 2 ifadesi ise ilgili yeteneği tek başına temsil etmede yetersiz kaldığı anlamına gelmektedir. Her bir ana ve alt yetenek ikilisi için bu şekilde bir ağırlıklandırma yapıldıktan sonra bu ağırlık değerleri algoritmada karşılaştırma işleminde kullanılmıştır. Matriste okunabilirliği kolaylaştırma amaçlı yetenekler sol sütuna, yetenekler ise tablo üst satırına alt yetenekler/ana yetenekler olacak şekilde yazılmıştır.

Çizelge 4.2. Niteleyici kelime - genel yetenek matrisi

	serbestanaliz/genel
Sistem Listesi	1
Alarm Listesi	1
Göndermeç Listesi	5
Karıştırma Analizi	10

Çizelge 4.2'den Karıştırma Analizi ifadelerinin Genel yetenekler alanında bulunan Serbest Analiz yeteneğini daha yüksek puanla temsil ettiği sonucu çıkarılmaktadır. "Göndermeç Listesi" ilgili yetenek için doğrudan bir ayırt edicilik oluşturmamaktadır. "Sistem Listesi" ve "Alarm Listesi" ifadeleri doğrudan ilgili yeteneği tek başına temsil etmede yetersiz kaldığı anlamına gelmektedir.



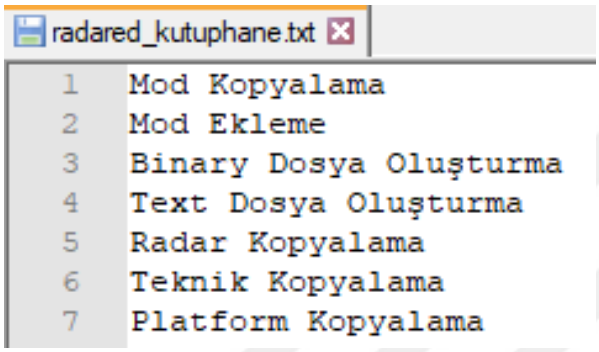
Şekil 4.2. Yeteneği temsil eden niteleyici ifadeler radar eh

Geliştiricilerle ortak yapılan çalışma sonrasında ana ve alt yetenekler sonrasında "senaryo, plan, kütüphane, genel, kaynak kontrolü, uçuş planlama" başlıklarına sahip, toplamda altı tane ana yetenek belirlenmiştir. Her bir ana yetenek başlığı altında belirlenen alt yeteneklerle birlikte ürün hattına yerleştirilebilecek potansiyelde için kırk beş alt yetenek belirlenmiştir. Her bir alt yeteneğin ve ona bağlı ana yeteneğin tespit edilebilmesi için niteleyici kelimeler oluşturulmuştur. Elde edilen tüm ifadeler Şekil 4.2'deki gibi txt uzantılı dosyaya kaydedilmiştir.

Şekil 4.2'de "Kütüphane" ana yeteneğine bağlı bir alt dal olan "Radar EH" alt aşlığındaki "kendini koruma" alt yeteneğini temsil eden ifadeler gösterilmektedir. Her bir yetenek dosyası alt yetenek ve ana yetenek ikilisi olacak şekilde isimlendirilmiştir. Doküman içerisinde bu kelimelerden birinin veya birkaçının geçmesi halinde o sistemde kullanılacak Görev Planlama ara yüzünde "Kütüphane" ana yeteneğine bağlı bir alt dal olan "Radar EH" alt başlığındaki "kendini koruma" alt yeteneği bulunması gerektiği anlaşılabilmektedir.

4.3. N-Gram Metodu ile Niteleyici Kelimelerin Düzenlenmesi

Sınıflandırma problemlerinde sıkça kullanılan ve verilen bir String dizisini daha alt parçalara ayırıp farklı karakterlerle ifade etmeyi sağlayan bir metot olan N-Gram yöntemini çalışmadaki probleme uyarlamak hedeflenmiştir. Bölüm 4.2'de anlatıldığı gibi her bir ana ve alt yeteneği temsil etmek için niteleyici kelime grupları oluşturulmuştur. Bu niteleyici ifadeler uzun tamlamalar halinde YGÖ dokümanı içerisinde geçebilmektedir. Şekil 4.3'te "Kütüphane" ana yeteneğindeki "Radar ED" grubunu temsil eden yetenekler verilmiştir.



Şekil 4.3. Yeteneği temsil eden niteleyici ifadeler radar ed

YGÖ içinde Şekil 4.3'teki ifadeler doğrudan geçmeyebilmektedir. Mesela doküman içinde "Binary Dosya Oluşturma" ifadesi yerine "Binary Dosya Gerçekleme" gibi bir anlatım yer alabilir. Bu diğer isim tamlamaları için de geçerlidir. Burada asıl olan anahtar kelimelere dikkat edilmelidir. Her bir niteleyici tamlamalarda yer alan kelimeler unigram, bigram, trigram olacak şekilde farklı kombinasyonlara ayrıştırılmıştır. Niteleyiciler doküman içinde hem unigram formlarıyla hem de daha fazla gruplar halinde aratılmıştır. Algoritma sonucunda 6 farklı çıktı elde edilmektedir. Ancak bu sonuçlardan hepsi "Kütüphane" ana yeteneği altındaki Radar ED için ayırt edicilik oluşturmamaktadır. Yukarıdaki örneğe göre oluşan 6 farklı çıktı "Binary", "Dosya", "Oluşturma", "Binary Dosya", "Binary Dosya Oluşturma" ve "Dosya Oluşturma" şeklinde olmaktadır. "Binary Dosya Oluşturma" niteleyici tamlamasında önemli olan kısım "Binary Dosya" ve "Binary Dosya Oluşturma" kelime gruplarıdır. Diğer tamlamaların yetenek için önemi yoktur, yeteneği temsil etmemektedirler. Bu yüzden algoritma çıktısıyla elde edilen tüm tamlamalar, her bir yetenek için önceden oluşturulmuş anahtar kelimeler dosyasıyla kıyaslanmaktadır. Anahtar kelime olup olmamasına göre ayırt etme işleminden sonra elde edilen çıktı Şekil 4.4'teki gibi oluşmaktadır.

```
key= radared_kutuphane.txt
*****************TAMLAMAAAA= Binary Dosya
*****************TAMLAMAAAA= Binary Dosya Oluşturma
*****************TAMLAMAAAA= Text Dosya
*****************TAMLAMAAAA= Text Dosya Oluşturma
*****************TAMLAMAAAA= Radar Kopyalama
*****************TAMLAMAAAA= Mod Kopyalama
*****************TAMLAMAAAA= Teknik Kopyalama
*****************TAMLAMAAAA= Platform Kopyalama
```

Şekil 4.4. Niteleyici tamlamalar

Şekil 4.4'te ilgili yetenek için sadece önemli olan kelime grupları yer almaktadır. Tamlamalar içinde yer alan "Radar", "Mod", "Text", "Binary" gibi tek kelimeler "Kütüphane" ana yeteneğinin Radar EH alt grubunu temsil etmemektedir. "Anahtar Kelimeler" dosyasına göre filtreden geçirilen tamlamaların doküman içinde aratılması sağlanmaktadır.

4.4. Yazılım Gereksinim Özellikleri Dokümanından Yetenek Çıkarımı

YGÖ dokümanları Türk Silahlı kuvvetlerinde geliştirilen projelerde yer alması istenen özelliklerin bir arada bulunduğu dokümanlardır. Belirli bir formata sahip olan bu dokümanlar yetenek listesi açısından sistemden sisteme farklılık göstermektedir.

Çalışmanın ikinci basamağında Türkçe olarak hazırlanmış pdf formatındaki YGÖ dokümanının parse edilmesiyle elde edilen kelimeler çıktı olarak alınmıştır. YGÖ dokümanından yetenek tespiti için Zemberek kütüphanesi ve PdfBox kütüphanesi kullanılmıştır. Her iki kütüphane içindeki metotlardan kelimelerin elde edilmesi ve işlenmesi sürecinde faydalanılmıştır. Şekil 4.5'te dizinde yer alan YGÖ dokümanının öncelikle metin olarak elde edilmesi için gerekli işlem yer almaktadır.

```
PDDocument document = PDDocument.load(new File("C:\\7.pdf"));
if (!document.isEncrypted()) {
    PDFTextStripper stripper = new PDFTextStripper();
    ygoText = stripper.getText(document);

}
```

Şekil 4.5. YGÖ dokümanının okunması

Dokümanın parse edilebilmesi için öncelikle Zemberek kütüphanesindeki "tokenize" metodu kullanılır. Bu metotla birlikte kullanılan "analyzeToken" metodu metin içinde yer alan ve Şekil 4.6'da gösterilen ifadeleri temizlemektedir.

```
static boolean analyzeToken(Token token) {
    return token.getType() != Type.NewLine
        && token.getType() != Type.SpaceTab
        && token.getType() != Type.UnknownWord
        && token.getType() != Type.RomanNumeral
        && token.getType() != Type.Unknown;
  }
```

Şekil 4.6. Dokümanın önişleme metodu

Kütüphanedeki "tokenize" metoduyla birlikte bazı kelimeler ayıklanırken bazıları değişime uğramaktadır. Çıktılar incelendiğinde Şekil 4.7'deki gibi bir değişim olmakta ve orijinal ifadeyi değiştirerek en yakın kelimeye dönüştürmektedir.

Elektronik Destek anlamına gelen ED kısaltması EN olacak şekilde değişmiştir. Bu durum, YGÖ dokümanı için sakıncalı durum oluşturmaktadır. Orijinal ifadenin değiştirilmesi yeteneğin tespit edilememesine yol açmaktadır. Bu yüzden değişime uğrayan önemli kelimelerin değişmeden diğer kelime gruplarının arasına katılmaları için işlem yapılmıştır.

```
|Correction: GPY -> GPS
|Correction: EKT -> ET
|Correction: ARN -> AN
|Correction: Ed -> En
```

Şekil 4.7. SuggestForWord metodu çıktıları

Değişime uğraması istenmeyen veya Zemberek kütüphanesinin algılayamadığı kelimeler için "özel kelimeler" dosyası oluşturulmuştur. "Göndermeç, Mod" gibi kelimeler oluşturulan yeni dosyaya yazılmıştır. Böylece Zemberek kütüphanesi denetimine uymayan kelimeler korunabilmiştir. Elde edilen kelime setlerinden, "asciiCozumle" metodu kullanılarak, "Kelime[]" dizileri oluşturulmuştur. Her bir kelime için "asciiCozumle" metodu uygulanmış ve kelimeler ek ve köklerine ayrıştırılmıştır. "asciiCozumle" metodu dışında "kelimeCozumle" metodu da kelimeleri ek ve köklerine ayrıştırmak için kullanılmaktadır.

Ancak araştırma ve test sonuçlarına göre “asciiCozumle” metodunun daha iyi performans verdiği gözlemlenmiştir. Çünkü “asciiCozumle” metodu kelimeleri mümkün olan köklerine ayırmakta ve bu kökler arasında yaygın kullanım frekansına göre sıralama yapmaktadır.

Yani bulunan ilk kökün kastedilen kelime olma olasılığı yüksek olmaktadır. EK 1 ve EK 5 arasında yer alan algoritmanın tamamından, özel kelimelerin dizinden okunuşu ve kelime dizilerinin elde edilme işlemleri Şekil 4.8’de yer almaktadır.

```
for(String mtn: metin) {
    if(zemberekSınıf.kelimeDenetle(mtn)){
        Kelime[] kelimeAsciiTolerans = zemberekSınıf.asciiCozumle(mtn);
        ygoKelimeleriAsciiTolerans.add(kelimeAsciiTolerans);
    }
}
ozelKelimelerDosyasi = dizindekiDosyayiGetir("ozelkelimeler.txt","C:\\");

BufferedReader br2 = new BufferedReader(new InputStreamReader(new FileInputStream(ozelKelimelerDosyasi), "UTF-8"));

String st2;
while ((st2 = br2.readLine()) != null) {
    ozelKelimeler.add(st2);
```

Şekil 4.8. Özel kelimelerin okunuşu ve kelime dizilerinin oluşturulması

Ana ve alt yetenekleri temsil edecek niteleyici ifadelerin txt uzantılı dosyadan okunarak algoritma içinde kullanılması sağlanmıştır. “AltYetenek_AnaYetenek” olarak isimlendirilen dosyalarda birden fazla sayıda kelime ve kelime grupları yer almaktadır. Örneğin “Senaryo” ana yeteneğin altında “almaç göndermeç” bölümünde iki liste vardır.

“Sistem” ve “Unsur” listeleri olarak iki listeye sahip olan “almaç göndermeç” alt yeteneği bir projede bulunacaksa, YGÖ içinde Sistem ya da Unsur listelerinin özellikleri bulunması gerekmektedir. Sistem ve Unsur listelerini betimleyecek ifadeler doküman içinde geçmelidir. Şekil 4.9’da “Sistem” listesi için niteleyici ifadeler yer almaktadır.

almacgondermec_sistem_senaryo.txt

```
1  Aldatma Listesi
2  Kestirme Listesi
3  Kaydetme Listesi
4  Almaç atama
5  Göndermeç Atama
6  Konum Atama
```

Şekil 4.9. Yeteneğin niteleyici ifadeleri

YGÖ dokümanında yer alan her bir kelime elde edildikten ve gereksiz ifadelerden temizlendikten sonra 4.2. Niteleyici Kelime Setlerinin Oluşturulması bölüm başlığında anlatılan yetenekleri içeren 45 adet dokümanın her biri sırasıyla işleme alınmıştır. Üç farklı yöntemle niteleyici ifadeler YGÖ içinde aratılmıştır. Öncelikle dizinde yer alan her bir doküman içinde yer alan niteletici ifadeler bütün olarak YGÖ içinde aratılmıştır. Sonra her bir niteleyici ifade kelimelere ayrıştırılmıştır ve kelime bazında aratma yapılmıştır. Bu tek kelimelerle aramada unigram metodu sonucu oluşan kelime çıktıları kullanılmıştır. En son olarak bu iki yöntemin harmanlanmasıyla oluşan kelime ve tamlama yöntemi kullanılmıştır. Son yöntemde bigram ve trigram metot çıktılarıyla oluşan tamlamalardan ve unigram çıktılarıyla oluşan kelimeler kullanılmıştır.

4.4.1. Tüm kelime aratma yöntemi

Yetenekleri temsil eden ifadelerin bulunduğu dokümanların tümü, kalıp olarak YGÖ içinde aratıldığında çıktı Şekil 4.10'daki gibi olmuştur. İfadeler kalıp olarak aratıldığında eşleşme sayısı daha az olmaktadır. Bunun sebebi bazı niteleyici ifadelerin YGÖ dokümanında farklı şekilde yer almasıdır. Mesela "Senaryo" ana yeteneğine bağlı Muhabere alt alanında yer alan yetenekleri temsil eden ifadelerden biri "Kaynak Kontrolü Yapma" olarak verilebilmektedir. YGÖ içinde bu yetenek "kaynak kontrolü yapabilecektir" olarak geçiyorsa tüm kelime ile arama yönteminde bu ifade göz ardı edilmektedir.

```
Tum Kelime Arama icin Eslesme Sayisi:33
eslesen: Nokta
eslesen: Polar
eslesen: Histogram
eslesen: Konum
eslesen: Yön
eslesen: Mesafe
eslesen: Yükseklik
eslesen: Bölge
eslesen: Hedef Listesi
eslesen: Alarm Listesi
...
```

Şekil 4.10. Tüm kelime aratma ile elde edilen çıktı

4.4.2. Kelime aratma yöntemi

Tüm kelime ile arama yöntemine rakip olarak kullanılan bu yöntemde, yetenekleri temsil eden bu niteleyici ifadelerin de kelime olarak ele alınıp ek ve köklerine ayrıştırılmasıdır. Böylece niteleyici ifadelerin birebir karşılığı olmasa bile farklı şekilde ifade edilmiştir.

Bu şekilde de muhtemel varyasyonlarının yakalanma şansı artmaktadır. Kelimelerin köklerine göre aratılması, farklılaşabilen eklerinden ayırt edilmelerini sağladığı için dokümanda gözlenme olasılığı yüksek olmaktadır. Yapım eklerinin korunması ve çekim eklerinin temizlenmesi sağlandıktan sonra her bir kelime için esas form elde edilebilmiştir.

Her bir nitelik kelimelere ayrıldıktan sonra sakıncalı yeni bir durumun oluştuğu gözlemlenmiştir. Bazı kelimelerin aratılmak istenen isim formatından farklılaştığı ve başka ek ve köklere dönüştüğü gözlemlenmiştir. Örneğin "Yayılım Kaybı" tamlaması için elde edilen kelime dizisi [yayılım, kayıp] olarak gözlemlenmiştir. Bu durumda YGÖ içinde sadece "yayılım" kelimesi yakalanabilmektedir. Başka bir örnekte ise "örüntü" ifadesinin "ör" ile değiştirildiği gözlemlenmiştir. Bu durumun önüne geçmek amacıyla, böyle değişime uğrayan kelimelerin kökü alınmadan olduğu gibi kullanılması sağlanmıştır.

YGÖ içindeki gereksinimlerde "Alarm Listelerinde silme işlemi yapılabilecektir" cümlesi geçebilmektedir. Kelimelere ayırma işlemi sonrasında "Alarm Listesi" şeklindeki bir kelime grubunda yer alan kelimeler "alarm" ve "listesi" olacaktır. Bu durumda "Listesi" kelimesinin aratılmasıyla örnek verilen cümledeki ifade gözden kaçacaktır. Bunun gibi sorunların önüne geçebilmek adına kelimeler çekim eklerinden temizlenmiştir. Yapım ekleri kelimeler yeni anlamlar kattıkları için onlara bir işlem uygulanmazken çekim eklerinin temizlenmesi ile orijinal kelimeye daha çok benzetme yapılmıştır. Bu şekilde yapılan geliştirme sonrasında Şekil 4.11'deki gibi bir çıktı elde edilmiştir.

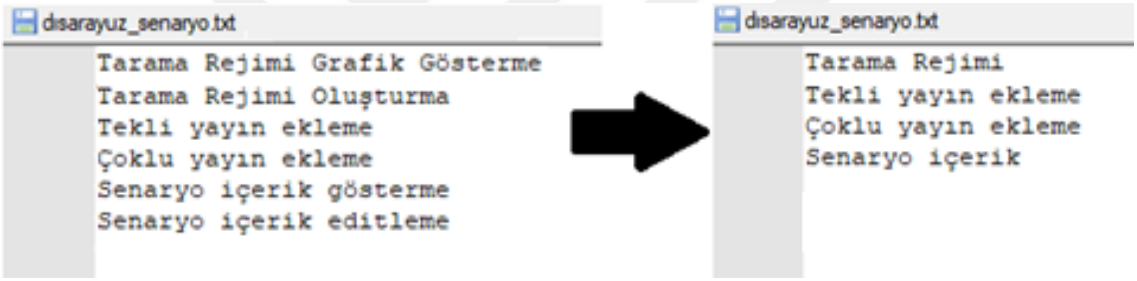
```
key= radaret_senaryo.txt
value= Platform, value= Silah,
key= muhabereeh_senaryo.txt
value= ilişki, value= Alarm, value= Dönme, value= Almaç, value= Baraj,
key= muhabereeh_sabityayin_senaryo.txt
value= Almaç,
```

Şekil 4.11. Kelime aratma ile oluşan çıktı bölümü

Kelimelerin her biri bağlı bulundukları yetenek dosyasını temsil etmektedir. YGÖ içinde aratılan ve rastlanan kelimelerle birlikte bağlı bulundukları dosyanın ismi yazdırılacak şekilde kodlama yapılmış ve Şekil 4.10'daki sonuç alınmıştır. Yapılan gözlemler sonucunda oluşan çıktıların tam olarak doğru sonuç vermediği gözlemlenmiştir. Şekil 4.11'de "Senaryo" ana yeteneğine bağlı Muhabere EH'yi temsil eden kelimeler "ilişki", "Alarm", "Dönme", "Almaç" ve "Baraj" olarak görüntülenmiştir.

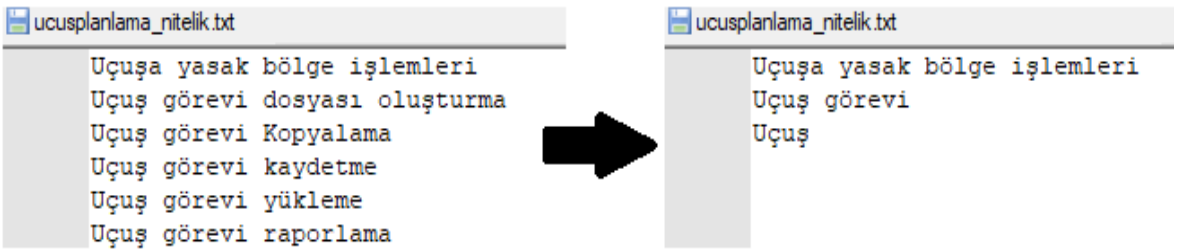
Oysaki “ilişki”, “alarm” ve “dönme” kelimeleri bir projede “Senaryo ve buna bağlı Muhabere yetenekleri var” diyebilmek için nitelik oluşturmamaktadır.

Bu tür sonuçların oluşmasını önlemek adına önceden oluşturulmuş “anahtar kelimeler” isimli dosya kullanılmıştır. Her bir yetenek dosyasının içinde bulunan ifadelerden hangilerinin önemli olduğunu anlayabilmek için bu dosya kullanılmıştır. Böylece gerçekten niteliği temsil edecek kelime ve kelime gruplarının elde edilmesi sağlanmıştır. Şekil 4.12’de sol tarafta bir yetenek dokümanı için oluşturulmuş niteleyici kelimeler ile sağ tarafta aynı yetenek dokümanını temsil eden anahtar kelimeler arasındaki fark gözlemlenebilmektedir. Anahtar kelimeler dosyasında niteleyiciler sadece kelime olarak yer almamaktadır. Kelime tamlamaları olarak da yer almaktadır. Sadece kelime bazlı arama yapıldığında hem YGÖ dokümanında eşleşme oranı düşük olmakta hem de anahtar kelimelerle kıyaslanması zor olmaktadır.



Şekil 4.12. Senaryo dış arayüz anahtar kelimelerin oluşturulması

Şekil 4.12’de “Senaryo” ana yeteneğinin “Dış Arayüz” alt yeteneğini temsil eden ifadelerin düzenlenerek daha sade bir hal aldığı gözlemlenmektedir. “Dış Arayüz” yeteneği için aslında görselin sağ tarafında yer alan kelime ve kelime gruplarının dokümanda rastlanması yeterli olmaktadır.



Şekil 4.13. Uçuş planlama anahtar kelimelerin oluşturulması

Benzer şekilde Şekil 4.13’de “Uçuş Planlama” yeteneği için niteleyici ifadelerde sadeleştirme yapılmıştır. Anahtar kelimeler sağ tarafta dosyadaki gibi düzenlenmiştir. Bu örneklerdeki gibi bir çalışma her bir yetenek dokümanı için yapılmıştır.

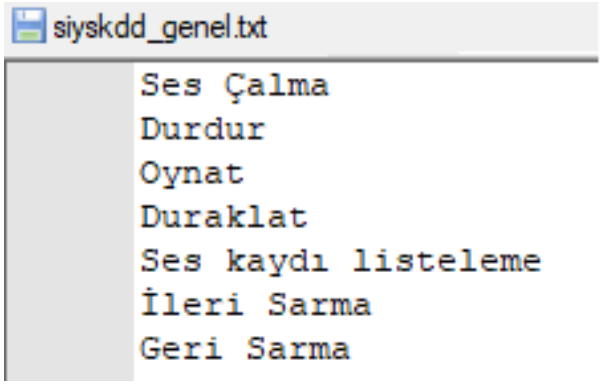
Bazı kelimeler tek başına bile etkili olabilmektedir. YGÖ dokümanında rastlandığında doğrudan projede bulunması gereken yeteneği temsil etmektedir. Bu tür kelimelere "harita", "grafik", "EOIR" ifadeleri örnek verilebilir. "harita" kelimesi net bir ifadedir ve projede harita bileşeninin olması gerektiğini temsil etmektedir. "grafik" kelimesi grafik bileşeninin olması gerektiğini ifade etmekte ve "EOIR" kelimesi elektro optik bileşeninin kullanılacağını temsil etmektedir.

Ancak çoğu kelime tek başına bir anlam ifade etmemektedir. YGÖ içinde geçiyor olması o yetenek bileşeninin kullanılacağı anlamına gelmemektedir. Bunun gibi yalnız başına anlamı olmayan kelimeler için bir öncesinde veya bir sonrasındaki kelimelerin de aratmaya katılması ve kelime tamlamaları oluşturularak aratma yapılması gerektiğine karar verilmiştir.

4.4.3. Kelime ve tamlama aratma yöntemi

Yetenek dokümanlarındaki niteleyici ifadelerin kelime olarak tek başlarına aratılmaları yanıltıcı sonuç verebilmektedir. Şekil 4.14'te "Ses" bileşeni için gerekli olan anahtar yetenekler gösterilmiştir. Bu yetenekler kelime olarak ayrıştırıldığında "Ses Kaydı Listeleme" ifadesinden "ses", "kayıt" ve "liste" kelimeleri elde edilmektedir. Buradaki "kayıt" ve "liste" kelimelerinin YGÖ içinde herhangi bir cümlede geçme şansı oldukça yüksektir. Örneğin YGÖ içinde "kayıtların analizi yapılabilecektir" cümlesi olması durumunda "kayıt" kelimesi bulunacaktır.

"Kayıt" kelimesi "Ses" bileşeni için niteleyici olarak yer aldığından dolayı "bu projede GPY yeteneği olarak "Ses" bileşeni kullanılacaktır" anlamı oluşmaktadır. Bu yüzden kelime eşleşmesiyle olan yöntem yeterli olmamaktadır. Burada tam olarak "Ses" bileşenini "ses kaydı" veya "ses kaydı listeleme" ifadeleri temsil etmektedir.



Şekil 4.14. Siy/Skdd bileşeni için anahtar kelimeler

Her bir yetenek dokümanı içindeki niteleyici kelime grupları, farklı tamlamalara dönüştürülmüş ve yeni formlarıyla aratılmıştır. Şekil 4.15'te niteleyici kelimelerden oluşturulan tamlamalar verilmiştir.

```
**********TAMLAMAAAA= Kayıt oynatım
**********TAMLAMAAAA= Ses Çalma
**********TAMLAMAAAA= Durdur
**********TAMLAMAAAA= Oynat
**********TAMLAMAAAA= Duraklat
**********TAMLAMAAAA= Ses kayıt
**********TAMLAMAAAA= Ses kayıt listeleme
**********TAMLAMAAAA= İleri Sarma
**********TAMLAMAAAA= Geri Sarma
**********TAMLAMAAAA= Ses kayıt filtreleme
key= siyskdd_genel.txt
```

Şekil 4.15. Ses bileşeni tamlamalarının örnek çıktısı

```
**********TAMLAMAAAA= Uçuşa yasak
**********TAMLAMAAAA= Uçuşa yasak bölge
**********TAMLAMAAAA= Uçuşa yasak bölge işlem
**********TAMLAMAAAA= Uçuş görev
**********TAMLAMAAAA= Uçuş görev dosya
**********TAMLAMAAAA= Uçuş görev dosya oluşturma
**********TAMLAMAAAA= Uçuş görev Kopyalama
**********TAMLAMAAAA= Uçuş görev kaydetme
**********TAMLAMAAAA= Uçuş görev yükleme
**********TAMLAMAAAA= Uçuş görev raporlama
key= ucusplanlama_nitelik.txt
```

Şekil 4.16. Uçuş planlama bileşeni tamlamalarının örnek çıktısı

Şekil 4.15'te ve Şekil 4.16'da "Ses" ve "Uçuş Planlama" bileşenleri için niteleyici kelime grupları verilmiştir. "Uçuş Planlama" bileşenini temsil etmesi için Şekil 4.13'te sağ taraftaki dosyada birçok niteleyici anahtar ifade yer almaktadır. Kelime tamlamaları oluşturulduktan sonra anahtar niteleyicilere yeni tamlama ifadeler de dahil edilmiştir.

Bu işlemler tüm yetenek dosyaları için yürütülmüştür. Böylece her bir yeteneği temsil etmesi için niteleyici ifadelerden anahtar özellikte olanlar seçilmiş ve anahtar olan niteleyici ifadelerin tamlamaları kullanılmıştır. Yeni algoritmada, YGÖ içinde bu anahtar nitelikteki kelime ve kelime grupları aratılmıştır. Bu sayede yeteneği daha doğru şekilde ortaya çıkaracak arama işlemi yapılabilmiştir. Yeni arama yönteminde ortaya çıkan sonuç Şekil 4.17'deki gibidir.

```
key= dısarayuz_radareh_kutuphane.txt
value= Kimliklendirme,
key= kutuphane_nitelik.txt
value= Kütüphane,
key= radareh_kendinikoruma_kutuphane.txt
value= Lazer, value= Lazer Tehdit,
key= radared_kutuphane.txt
value= Platform, value= Silah,
value= Platform liste, value= Silah liste,
key= radareh_radaret_kutuphane.txt
value= Hareket, value= Profili,
value= Hareket Profili
--------------------
key= grafik_genel.txt
value= Grafik,
key= listeyetenekleri_genel.txt
value= dışa, value= içe, value= arama,
key= harita_genel.txt
value= Konum, value= Bölge, value= Yön,
value= Yükseklik, value= Mesafe,
key= reform_genel.txt
value= Sürüm, value= şifre, value= Yardım,
value= Tema, value= Dil, value= Terimler, value= Hakkında,
```

Şekil 4.17. YGÖ içinde tespit edilen kelime grupları ve temsil ettikleri yetenekler

Şekil 4.17'de tüm yetenekler için oluşturulan kelime gruplarının YGÖ içinde aratıldıktan sonraki çıktısının bir kısmı yer almaktadır. Çıktıya göre "key" olarak belirtilenler ana ve alt yetenek setlerinin ismini göstermektedir. "value" olarak belirtilenler ise bağlı bulunduğu yeteneğin tespit edilmesinde görev alan kelime ya da kelime grubunun isimlerini göstermektedir. Seçilen proje için "Grafik", "Liste", "Harita", "Reform" bileşenlerinin ve "Kütüphane" ana bileşeninden "Radar ED", "Radar EH" ve "Dış Arayüz" alt bileşenlerinin olduğu anlamı çıkarılabilmektedir. Ayrıca "Radar EH"nin kendi içinde ayrılan alt kollarında "Radar ET" ve "Radar ED"nin de projede yer alacağı bilgisi çıkarılmaktadır.

Ana ve alt yetenekleri temsil eden ifadeler dışında elde edilen çıktılardan bir diğeri de Şekil 4.18'de verildiği ana ve alt yeteneklerin listesidir. Önce ana yetenek sonrasında da ana yeteneğe bağlı alt yetenekler olacak şekilde alt alta yazdırılacak şekilde bir çıktı oluşturulmuştur. Bu şekilde kullanılan çıktının yazılım geliştiricisi tarafından okunabilirliği çok daha yüksek olmaktadır.

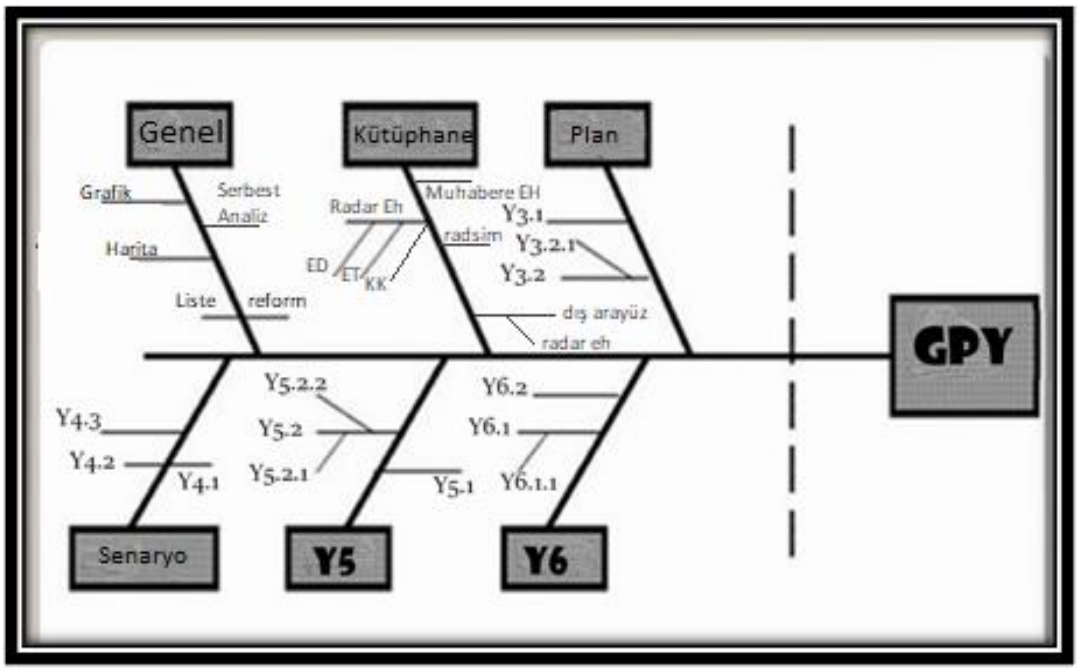
```
key= nitelik.txt
value= kutuphane,
value= plan,
value= senaryo,

key= genel.txt
value= grafik,
value= harita,
value= listeyetenekleri,
value= reform,
value= serbestanaliz,

key= kutuphane.txt
value= disarayuz, value= radareh,
value= muhabere,
value= radared,
value= radareh, value= kendinikoruma,
value= radareh, value= radared,
value= radareh, value= radaret,
value= radaret,
value= radsim,
```

Şekil 4.18. Ana ve alt yetenekler

Şekil 4.18'de çıktının bir kısmı verilmiştir. "Nitelik" başlığında projede bulunacak ana yetenekler yer almaktadır. Sonrasında her bir ana yeteneğin ismi ve alt kısımlarında bu ana yeteneklere bağlı alt yeteneklerin isimleri yer almaktadır. Örnekte kütüphane ana yeteneğine bağlı bulunan alt yetenekler ve alt yeteneklerin de bir alt yetenekleri listelenmiştir. Yeni projelerin algoritmaya koyulan yeni çıktılarına göre ürün hattına uygun eklemeler yapılarak süreç devam ettirilmiştir. Ürün hattına yerleştirilen yeteneklerin küçük bir kısmının örneği Şekil 4.19'da gösterildiği gibi yapılmıştır.



Şekil 4.19. Yeteneklerin ürün hattına yerleşim örneği

Çalışmada öncelikle proje 1 için YGÖ dokümanı incelenmiştir. İnceleme sonunda pek çok gerekli olmayan kelimenin de dokümanda eşleştiği gözlemlenmiştir. Mesela “Radar”, “Elektronik”, “Sistem” kelimeleri bağlı bulunan sektörün isminde geçtiği için YGÖ içinde de sıkça kullanılmaktadır. Bunun gibi “Radar” kelimesiyle benzer şekilde “liste”, “sürüm”, “veri” gibi kelimeler doküman içinde farklı şekillerde pek çok kez geçebilmektedir. Kelimeler, yetenek belirtmediği için tek başlarına kullanılmalarının anlamsız olduğu gözlemlenmiş ve sık kullanılan bu kelimelerin elenmesi sağlanmıştır. Bu eleme işlemi yapılmadan önce hemen hemen her yeteneği temsil eden bir niteleyici gibi işlem görmüş ve sonuçların yanıltıcı olmasına yol açmıştır. Proje başında belirlenen tüm alt yeteneklerin çıktı olarak alındığı gözlemlenmiştir. İlerleyen aşamalarda YGÖ içindeki kelimelerin değişime uğradığı ve niteliklerle eşleştirilemediği gözlenmiştir. Örneğin GPY kelimesinin GPS olarak değiştirilmesi yanılgılara yol açmaktadır. Bu tür kelimeler için “özel kelimeler” listesi oluşturulmuş ve orijinal hallerinin korunması sağlanmıştır.

Çalışmanın devamında diğer projelerin YGÖ dokümanları da eğitim verisi olarak algoritmaya verilmiştir. Tamlama kelimelerin oluşturulmasıyla ve içlerinden anahtar nitelikte olanların seçilmesiyle oluşturulan kelime dizileri kullanılmıştır. Yeni kelime gruplarıyla birlikte oluşan çıktılar değerlendirilerek anahtar kelimeler listesinde güncellemeler yapılmıştır. Algoritmanın yapılan iyileştirmelerle daha başarılı sonuç verdiği gözlemlenmiştir.

Çalışmaların devamında elde edilen çıktılarda dokümanda gözlemlenen her bir niteleyici kelime grubuna göre ana yetenek ve alt yeteneklerin tespit edildiği gözlemlenebilmektedir. Şekil 4.20’de örnek bir çıktının bir parçası yer almaktadır. Çıktıya göre “Uçuş Planlama” ana yeteneği altında patika belirleme ve patika doğrulamaya yönelik iki alt yetenek gözlemlenmektedir.

“Genel Yetenekler” gurubundan Dış Sistem, Grafik, Harita, Listeleme, Reform, Serbest Analiz ve Ses İşleme yetenekleri olduğu görülebilmektedir. “Kütüphane” ana yeteneği altında ise Dış Arayüz, Muhabere ve Radar EH olduğu tespit edilmiştir. Çıktılarda Muhabere EH’nin alt dalı olan Radar ED yeteneği ve Radar EH’nin iki alt dalı olan Kendini Koruma ile Radar ED yetenekleri de gözlemlenebilmektedir. Her bir ana yeteneğe bağlı alt yetenek gözlemlenebildiği gibi alt yeteneklere bağlı bir alt yetenek daha mevcut ise bu yetenekler de tespit edilebilmektedir.

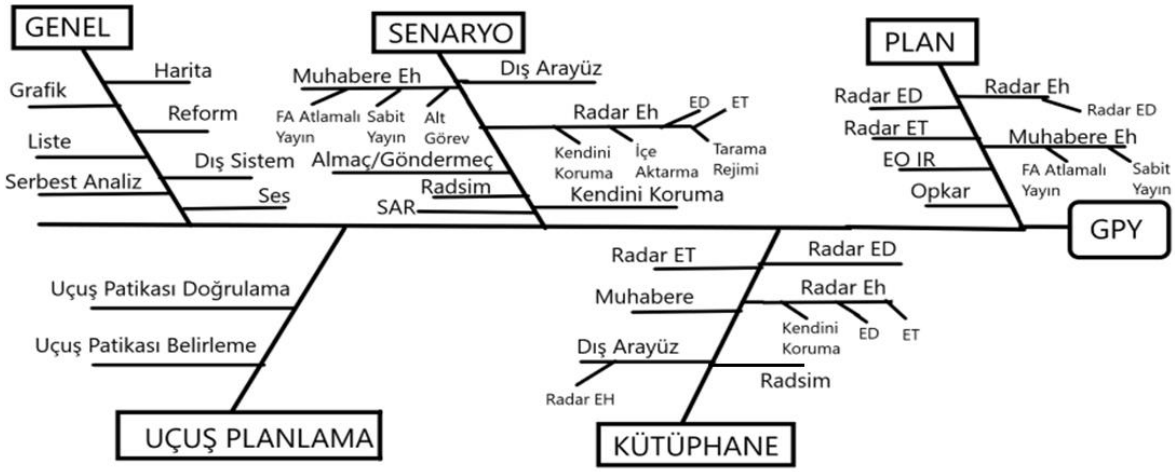
```
key= ucusplanlama.txt
value= ucuspatikasibelirleme,
value= ucuspatikasidogrulama,

key= genel.txt
value= dissistem,
value= grafik,
value= harita,
value= listeyetenekleri,
value= reform,
value= serbestanaliz,
value= siyskdd,

key= kutuphane.txt
value= dısarayuz,
value= radareh,
value= muhabere, value= radared,
value= radareh, value= kendinikoruma,
value= radareh, value= radared,
```

Şekil 4.20. Tespit edilen ana ve alt yetenekler

Elde edilen çıktılar en başta ana yetenek alt kısmında iste alt yetenek ve varsa ona bağlı alt yetenekler olacak şekilde oluşturulmuştur. Tüm bu sonuçlar diğer proje çıktılarıyla birleştirilerek ürün hattına yerleştirilmiştir. Gözlemlenen tüm yeteneklerle Şekil 4.21'deki ürün hattı oluşturulmuştur.



Şekil 4.21. GPY ürün hattı

5. PERFORMANS DEĞERLENDİRMESİ

YGÖ dokümanlarının sayısı proje sayısına bağlı olduğu için literatürdeki sınıflandırma algoritmaları gibi büyük ölçekli veri seti kullanılamamıştır. Tüm dokümanların %60'ı eğitim ve %40'ı da test verisi olacak şekilde kullanılmıştır. Çalışma performansının değerlendirilmesi iki farklı kriter ile gerçekleştirilmiştir. Bu kriterler problemin ortaya çıkış sebebi göz önünde bulundurularak belirlenmiştir. Yeteneklerin tespit edilme doğruluğu ile tespit edilme süreleri ölçülmüştür. İlk değerlendirme algoritmanın başarılı sonuç vermesine göre yapılmıştır. Bu süreçte algoritmadan çıkan sonuçlar ile öncesinde gözle inceleme sonucu çıkarılmış yetenek kayıtları incelenmiştir. Geçmiş zamanlarda ve önceki projelerde YGÖ dokümanının manuel olarak geliştirici tarafından incelenmesiyle ortaya çıkan proje incelenmiştir. Projedeki ara yüzlerde yer alan ana ve ara yetenekler kayıt altına alınmıştır. Sonrasında algoritmadan çıkan sonuçlar ile bu kayıtlı yeteneklerin benzerliği ölçülmüştür. Çizelge 5.1'de her iki çıktının da bir parçasının karşılaştırılması mevcuttur.

Çizelge 5.1. Proje 1 için oluşan çıktıların karşılaştırılması

Algoritma Çıktı Sonucu	Kütüphane Ana Yeteneği Listeleri
key= dısarayuz_radareh_kutuphane.txt value= kimliklendirme, value= Manuel Kimliklendirme	Radar Listesi
	Platform Listesi
key= kutuphane_nitelik.txt value= Kütüphane,	Silah Listesi
	Radar-Platform Listesi
key= radareh_radaret_kutuphane.txt value= Hareket, value= Profili, value= Hareket Profili, value= EKT liste, value= Teknik liste,	Teknik Listesi
	Radar-Teknik Listesi
key= radaret_kutuphane.txt value= Platform, value= Silah, value= Platform Liste, value= Silah Liste,	Hareket Profili Listesi

Çizelge 5.1'deki tablonun sağ tarafında projede "Kütüphane" ana yeteneği altında bulunan listeler verilmiştir. Bu listeler Görev Planlama ara yüzü incelenerek kayıt altına alınmıştır. Tablonun sol tarafında ise algoritma çıktısının kütüphane yeteneklerinin kapsayan kısmı yer almaktadır. Çıktının sonucuna göre "kutuphane_nitelik" ifadesi projede kütüphane yeteneği olduğunu göstermektedir.

“dısarayuz_radareh_kutuphane” ifadesi alt yetenek olarak “Dış Ara yüz” ve buna bağlı Radar EH olduğunu temsil etmektedir. “radareh_radaret_kutuphane” ifadesi “Kütüphane” ana yeteneğine bağlı Radar EH alt yeteneğinden Radar ET’nin projede yer aldığını göstermektedir. Benzer şekilde “radaret_kutuphane” ifadesi “Kütüphane” ana yeteneğine bağlı Radar ET’nin projede yer aldığını göstermektedir. Her yetenek altında verilen “value” değerleri ise o yeteneğin tespit edilmesinde kullanılan anahtar kelimeyi temsil etmektedir. Bu anahtar kelimeler YGÖ dokümanı içinde yer aldığı için yetenek tespiti yapılabilmiştir. Elde edilen çıktının tamamı EK 8 ve 9’da yer almaktadır. Çıktının devamında Şekil 5.1’deki yetenekler yer almaktadır.

```
--------yetenekler-----------
key= nitelik.txt
value= kutuphane, value= plan, value= senaryo,

key= plan.txt
value= muhabereeh, value= frekansatlamaliyayin
value= muhabereeh, value= sabityayin,
value= radared,
value= radareh, value= radared,

key= senaryo.txt
value= radareh, value= kendinikoruma,
value= radareh, value= radared,
value= radareh, value= radaret,
value= radareh, value= taramarejimi,

key= genel.txt
value= grafik, value= harita,
value= listeyetenekleri, value= reform,
value= serbestanaliz,

key= kutuphane.txt
value= dısarayuz, value= radareh,
value= radareh, value= radaret,
value= radaret,
```

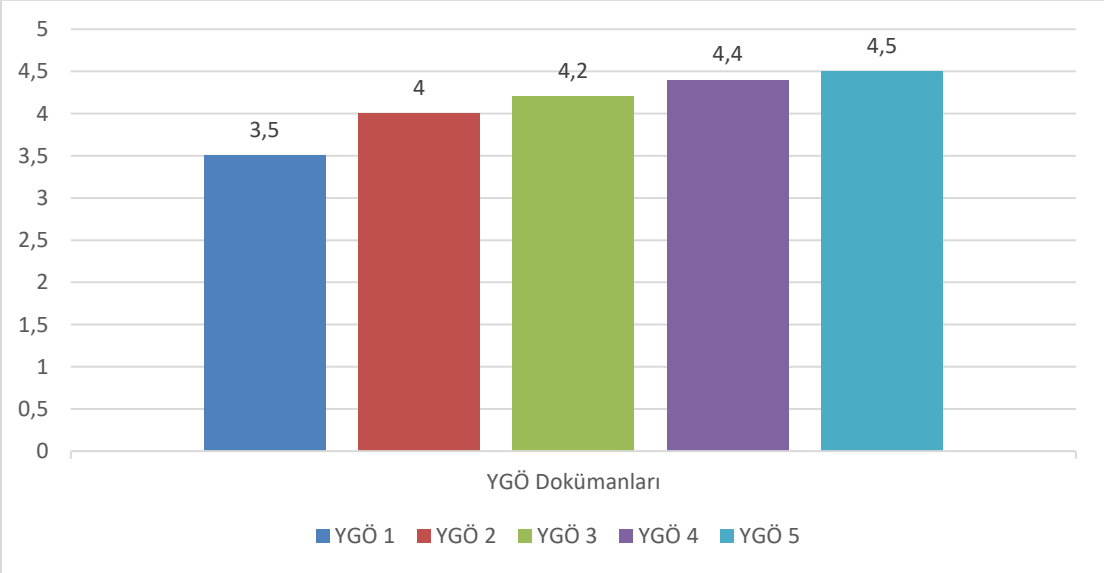
Şekil 5.1. Tespit edilen ana ve alt yetenekler

Şekil 5.1’de sisteme verilecek Görev Planlama yazılımında yer alacak ana yetenekler “key” olarak belirtilmiştir. Her bir ana yeteneğe bağlı alt yetenekler ise “value” olarak verilmiştir. En üstte yer alan “nitelik” ifadesi altında ana yetenekler yer alırken sonrasında ana yetenekler ve onlara bağlı alt yetenekler isim olarak verilmiştir. “Genel” yetenekler altında “Grafik, Harita, Liste, Reform ve Serbest Analiz” yetenekleri mevcutken “Plan” ana yeteneği altında “Muhabere EH, Radar EH ve Radar ED” alt yetenekleri mevcuttur.

Bu alt yeteneklerin de alt yetenekleri Frekans Atlamalı Yayın, Sabit Yayın gibi yetenekler yine Şekil 5.1'de gözlemlenebilmektedir. "Senaryo" ana yeteneği altında sadece Muhabere EH alt yeteneği olduğu ve buna bağlı farklı alt dalların varlığı gözlemlenebilmektedir.

Algoritma sonucu elde edilen yeteneklerin önceden oluşturulmuş gerçek yeteneklerle olan benzerliği yapılan iterasyonlar sonucunda daha başarılı sonuçlar vermiştir. Her bir YGÖ dokümanında elde edilen veriler daha iyi yorumlanmış ve gerçek yeteneklerin tespitinde doğruluk oranının arttığı Şekil 5.2'de grafiksel olarak verilmiştir.

Başarı oranının artmasının sebebi her gereksinim dokümanıyla yeni ifadelerin yakalanabilmesi ve yeni niteleyicilerin eklenmesidir. Benzer şekilde bir niteleyici eksik veya yanlışsa yapılan testler sonucunda bu durum düzeltilmekte ve algoritmaya daha doğru girdiler sağlanmaktadır. Önceden oluşturulan niteleyici ifadelerle ilk YGÖ dokümanı test edildiğinde başarı %70 civarındadır. Çıkan sonuçlara göre niteleyici ifadelerin düzenlenmesinin ardından başarı oranı her yeni YGÖ ile artmıştır. Denemesi yapılan ve başarımın arttığı gözlemlenen ilk beş YGÖ dokümanı sonuçları grafikte verildiği gibidir. Sonrasında kullanılan YGÖ dokümanlarında yetenek tespiti benzer şekillerde başarı göstermektedir.



Şekil 5.2. Yetenek tespitinin doğruluk oranlarının karşılaştırılması

Yeteneklerin büyük oranda benzerliği sağlandıktan sonra diğer performans kriteri olarak "yetenek tespit zamanı" ele alınmıştır. Her bir projenin YGÖ dokümanının incelenerek projede yer alacak yeteneklerin tespit edilmesi hemen hemen birbirine benzer zaman almaktadır. Çizelge 5.2'de projelere göre yetenek tespiti süreleri verilmiştir.

Bu tespit süreleri iki şekilde ölçülmüştür. İlk olarak otomatik olarak yetenek çıkarılmasından sonra gözle hızlıca bir gözden geçirme ile yeteneklerin kontrol edilmesi ve son halinin alınmasına kadar geçen süre ölçülmüştür. İkinci olarak YGÖ dokümanının tamamının gözle incelenmesi ve yeteneklerin ortaya çıkarılmasına kadar geçen süre ölçülmüştür. Ölçüm sonuçları tablo halinde verilmiştir.

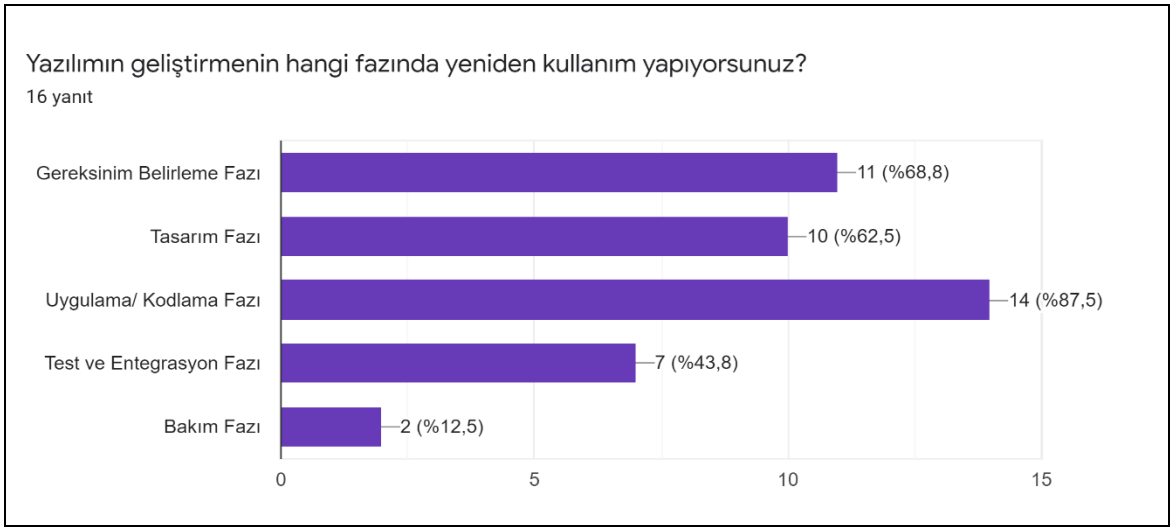
Çizelge 5.2. Yetenek Tespiti Sürelerinin Karşılaştırılması

	Otomatik Yetenek Tespiti (sn)	Yetenek Tespiti Sonrası Gözden Geçirme Süresi (dk)	Manuel Yetenek Tespiti Süresi (dk)
Proje 1	65	16	34
Proje 2	58	14	30
Proje 3	57	14	30
Proje 4	47	10	35
Proje 5	70	18	40
Proje 6	61	16	40
Proje 7	55	15	30
Proje 8	55	15	35

Tabloda farklı projeler için farklılık gösteren YGÖ dokümanının gözle incelenmesinin çok daha uzun sürdüğü gözlemlenmektedir. Algoritma ile elde edilen çıktıların manuel yetenek tespitine göre çok daha performanslı olduğu açıkça görülebilmektedir.

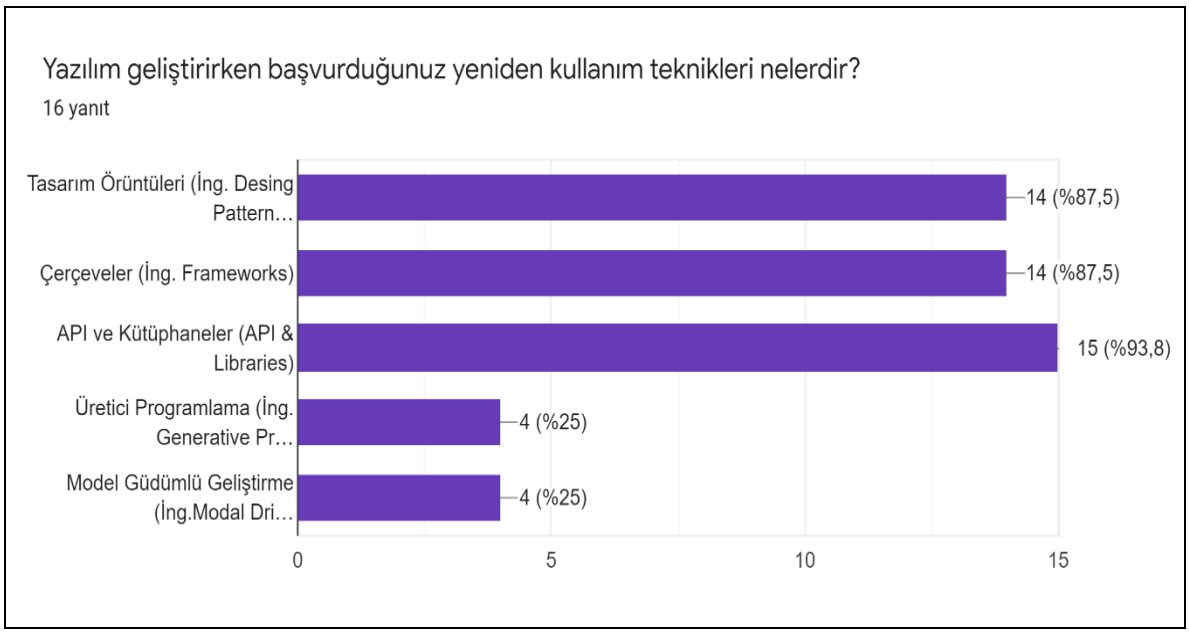
Yapılan çalışma çıktıları kullanılarak sonrasında daha kısa bir gözden geçirme ile projenin geliştirme ve teslimat süreçlerinde azalma sağlanmıştır. Süreç sonunda zamandan büyük ölçüde tasarruf sağlanarak hem yetenek tespiti yapılmış hem de Görev Planlama yazılımının yetenekleriyle ürün hattı oluşturulmuştur. Çalışmanın değerlendirme sürecinde EK 14'te sunulan anket çalışmasıyla tez kapsamında yapılan çalışmayı kullanacak geliştirici ekip yorumları alınmıştır.

Çeşitli pozisyonlardaki 16 kişiye yapılan anket sonucunda elde edilen sonuçlar EK 15'te detaylı olarak verilmiştir. Anket sonuçlarında OSGI yapısının sıklıkla kullanıldığı ve yeniden kullanımın önemli bir süreç olduğu açıkça gözlemlenebilmektedir. Anket içinde tezde uygulanan yöntem kısaca özetlenmiştir ve bu yöntemin uygulanabilirliği sorgulanmıştır. Şekil 5.3'te yeniden kullanımın yazılım geliştirmede hangi aşamalarda yer aldığı görülebilmektedir. En çok uygulama ve kodlama sürecinde kullanılmaktayken diğer geliştirme süreçlerinde de aktif rol oynamaktadır. En az ise bakım fazında kullanım alanı bulunmaktadır.



Şekil 5.3. Anket sonuçları yeniden kullanım

Anketi cevaplayanların çoğunlukla geliştirici ekip olması kodlama fazında yeniden kullanım oranını arttırmıştır. Yeniden kullanım işlemi ikinci olarak gereksinim belirleme sürecinde kullanılmaktadır. Bu durum projeler arasında yeteneklerin benzerlik gösterebildiğini ve ürün hattındaki yeteneklerin projeler arasında rahatlıkla kullanılabileceğini göstermektedir.



Şekil 5.4. Anket sonuçları yeniden kullanım teknikleri

Şekil 5.4'te kodlama sürecinde en sık API ve kütüphaneler kullanılmaktadır. Çerçeveler ve tasarım örüntüleri ise ikinci sırada yer almaktadır. Model güdümlü geliştirme ve üretici nitelikte programlama az bir paya sahiptir.

Şekil 5.5'te yapılmış olan çalışmanın değerlendirmesi yapılmıştır. Çalışmanın başarılı çıktı verebilmesi için yapılan işlemler ve süreç anlatılmıştır. Çeşitli pozisyonlardaki üyelerin değerlendirmeleri farklı olmuştur.

12 kişi orta kolaylıkta bulurken, 3 kişi zor ve 1 kişi de çok kolay olarak nitelendirmiştir. Anket'in genel değerlendirmesine göre tez çalışmasının uygulanması kolay bulunmuştur.

Şekil 5.5. Anket sonuçları uygulamanın kolaylık derecesi

6. SONUÇ VE ÖNERİLER

Tez çalışması kapsamında Görev Planlama yazılımı için YGÖ dokümanlarından otomatik yetenek tespiti ve ürün hattı oluşturma çalışmaları yapılmıştır. Çalışma öncesinde metrik-soru-hedef yaklaşımıyla çalışmanın sınırları belirlenmiştir. Gerçekleştirdiğimiz projeler kapsamında ihtiyaç analizi yapılmıştır. İhtiyaç analizi ve girdi setleri Görev Planlama ekibi üyelerince birlikte belirlenmiştir. Proje yeteneklerinin uzun zamanda, meşakkatli yollarla tespit edildiği saptanmıştır. YGÖ dokümanları gizlilik içerdiği için geliştirme çalışmaları ve testleri şirket içinde yürütülmüştür. Dokümanların gizlilik dereceli bilgi içermeleri, doküman formatları ve Türkçe diliyle oluşturulmuş olmaları proje süresinde yer yer zorluklara yol açmıştır. Gizli olmalarından ve dışarıya çıkarılamamalarından dolayı çalışmaların bir kısmı şirket içinde yürütülmüştür. Doküman içinde yer alan pek çok kalıp ifade, yetenek kelimeleriyle benzerlik gösterdiği ve içeriği olmayan başlıklar bulunduğu için ham bilginin elde edilmesinde de zorluklar yaşanmıştır.

Bunların yanı sıra girdi olarak kullanılan YGÖ dokümanı sayısındaki azlık da eğitim verisinin az olmasına ve ürün hattı yeteneklerindeki dallanmaların az olmasına yol açmaktadır. Daha çok doküman ile potansiyel yetenekler ve yetenek tespiti başarımları çok daha yüksek olacaktır. Giriş seti azlığından ötürü klasik makine öğrenmesi yaklaşımları uygulanamamış ancak literatürdeki diğer çalışmaların birleştirilmesiyle ortaya bir algoritma çıkarılmıştır. Literatürün incelenmesi sonrasında Türkçe dilinde yapılan sınıflandırma çalışmalarının fazla olmadığı gözlemlenmiştir. Çalışmaların çoğu büyük veri tabanları kullanılarak yürütülmüştür. Türkçe diliyle ilgili çalışmaların ortak noktası Zemberek kütüphanesinin kullanımı ve N-Gram yönteminin tercih edilmesidir. Tez çalışması sürecinde Zemberek ile elde edilen kelimeler YGÖ dokümanı yapısına göre düzenlenmiştir. Kelimelerin tamamen kök hallerinin veya orijinal tamlama hallerinin kullanılmasıyla birlikte yetenek tespit başarı oranının düşük olduğu gözlemlenmiştir. İlerleyen aşamalarda N-Gram yöntemi kullanılarak kelime grupları farklı kombinasyonlarla birleştirilerek unigram, bigram ve trigram yapılar oluşturulmuştur. Çekim eklerinden temizlenerek oluşturulan yeni kelime tamlamalarıyla YGÖ dokümanı içinde eşleşen yeteneklerin daha doğru olduğu gözlemlenmiştir. Yetenekler elde edildikçe ürün hattına hiyerarşik yapısı bozulmadan eklenmiştir.

Çalışmanın testi için otomatik çıkarılan yeteneklerin gerçek yeteneklerle örtüşme durumu ve doğruluğu geliştirici tarafından gözle yapılmıştır.

Sonrasında proje yeteneklerinin otomatik çıkarılma süresi ile manuel çıkarılma süresi arasındaki fark sayısal olarak gözler önüne konmuştur. Sayısal metriklerin yanı sıra yoruma dayalı başarı kriteri de anket çalışması sonrasında ölçülmüştür.

Yeteneklerin tespit edilmesinden sonra iki risk tespit edilmiştir. Bu risklerden ilki elde edilen sonuçlarda ana ve alt yetenekleri temsil eden niteleyici ifadelerin benzerliğinden dolayı YGÖ içinde tespit edilen bir nitelik, birden fazla yeteneği temsil edebilmektedir. Bu durumda projede yer alması gerekmeyen yeteneğin de varmış gibi görünmesine yol açabilmektedir. Bunun önüne geçebilmek için yetenekleri temsil eden niteleyici ifadeleri farklı şekilde ifade etme yöntemleri kullanılabilir. Gözlemlenen risklerden diğeri ise YGÖ içinde ilgili başlığın olması ama içeriğinin boş olmasıdır. Mesela "Kütüphane Kopyalama" başlığının doküman içinde yer alması ancak içeriğinin boş bırakılması yani o yeteneğin bulunmaması durumu oluşabilmektedir. YGÖ dokümanlarında boş başlıkların silinmesi sağlanabilirse bu durum da büyük ölçüde önlenmiş olacaktır. Dokümanda olan eksiklik veya fazlalıklar da algoritma çıktısını etkileyebilmektedir. Dokümanların öncesinde düzenlenmesiyle ve yetenekleri temsil edecek niteliklerin farklılaştırılmasıyla çok daha doğru sonuçlar elde edilecektir.

Tez çalışması sonunda yeteneklerin çıkarılmasının, hangilerinin ortak hangilerinin projeye özgü olduğunun belirlenmesi ve ürün hattı tasarımı yapılmıştır. Çalışmayla birlikte YGÖ dokümanlarında eksiklikler ve fazlalıklar olabileceği gözlenmiştir. Manuel yolla yetenek tespiti daha uzun zaman alan ancak daha güvenilir bir yöntemdir. Manuel yöntemde YGÖ dokümanındaki eksiklik veya fazlalıkların da daha rahat üstesinden gelinebilmektedir. Otomatize edilmiş bir sistemde bu süreç test sonuçlarıyla birlikte ispatlanarak çok daha kısa sürede yürütülmüştür. Algoritmanın %100 performanslı çalışması yeteneklerin tam belirlenmiş ve YGÖ dokümanı doğruluğuna bağlıdır. Algoritma sonrası oluşan çıktılar bilir kişilerce kontrol edilerek manuel yönteme kıyasla daha başarılı sonuçlar elde edilmiştir. Yetenek tespiti ve proje geliştirim sürecini büyük ölçüde kısaltarak geliştiriciye yardımcı olan bu çalışma sonunda Görev Planlama yazılımları kapsamında ürün hattı oluşturulmuştur.

6.1. Gelecek Çalışmalar

Tez kapsamında yapılan çalışma başlangıç olarak "Görev Planlama" yazılımı için yapılmıştır. Görev Planlama yazılımında yer alan yetenekler ve niteleyici ifadeleri oluşturulmuştur.

Sadece GPY özelinde olmayan harita, grafik, genel gibi yetenek başlıkları da YGÖ dokümanlarında mevcuttur. Önce GPY özelinde ana ve alt yetenekleri çıkarılan ve yetenek hattı oluşturulan çalışma, ilerleyen zamanlarda başka analiz projelerine de rahatlıkla uyarlanabilir niteliktedir. Algoritma girdisi olan YGÖ doküman sayısı arttırılarak literatürde yer alan doğal diş işleme teknikleri çalışma içine dahil edilebilir.

Buna ek olarak çalışmanın devamında kelime doküman matrisi veya yetenek doküman matrisi oluşturularak bir dokümanda yer alan yetenekler göz önüne alınabilir. Sonrasında yeni gelen bir projeye ait yeni dokümanların daha önceki hangi projedeki dokümanlarla benzerlik gösterdiği incelenerek projeler arası benzerlik çalışması ortaya konabilir. Artan veri setiyle birlikte makine öğrenmesi teknikleri dahil edilerek bir sonraki projenin hangi proje benzeyip hangi yetenekleri barındıracağı tespit edilebilecektir. Bu durumun geliştirme süreçlerine de katkısı yüksek olacaktır.

KAYNAKLAR

1. Sandhu, P., and Singh, H. (2006). *A Neuro-Fuzzy Based Software Reusability Evaluation System with Optimized Rule Selection*, International Conference on Emerging Technologies, 664–669.

2. Torkamani, M. A. (2014). Metric Suite to Evaluate Reusability of Software Product Line, *International Journal of Electrical and Computer Engineering*, 4 (4), 285-294.

3. Zhang, T., Deng, L., Wu, J., Zhou, Q., and Ma, C. (2008). *Some Metrics for Assessing Quality of Product Line Architecture*, International Conference on Computer Science and Software Engineering, 500-503.

4. Korra, S., Viswanadha, S. and Babu, A. V. (2013). Strategies for Designing and Building Reusable Software Components, *International Journal of Computer Science and Information Technologies*, 4 (5), 655-659.

5. Caldiera, G., and Basili, V. (1991). Identifying and Qualifying Reusable Software Components, *Institute of Electrical and Electronics Engineers*, 24 (2), 61-70.

6. Younoussi, S., and Roudies, O. (2015). All About Software Reusability: A Systematic Literature Review, *Journal of Theoretical and Applied Information Technology*, 76 (65), 64-73.

7. Babu, G.N.K. S., and Srivatsa, S.K. (2009). Analysis and Measures of Software Reusability, *International Journal of Reviews in Computing*, 1 (1), 41-45.

8. Torkamani, M.A. (2014). Metric Suite to Evaluate Resusability of Software Product Line, *International Journal of Electrical and Computer Engineering*, 4(2), 285-294.

9. Washizaki, H., Yamamoto, H., and Fukazawa, Y. (2003). *A metric süite for measuring reusability of software components*, 5th International Workshop on Enterprise Networking and Computing in Healthcare Industry, 211-223.

10. Atkinson, C., and Muthig, D. (2002). *Enhancing Component Resability through Product Line Technology*, International Conference on Software Reuse, 93-108.

11. Davidov, D., Tsur, O., and Rappoport, A. (2010). *Enhanced Sentiment Learning Using Twitter Hashtags and Smileys*, In Proceedings of COLING'10, 23rd International Conference on Computational Linguistics, 241-249.

12. Taboada, M., Brooke, J., Tofiloski, M., Voll, K., and Stede, M. (2011). Lexicon-based Methods for Sentiment Analysis, *Computational Linguistics*, 37 (2), 267-307.

13. Khan, F.H., Qamar, U., and Bashir, S., (2016). eSAP - A decision support framework for enhanced sentiment analysis and polarity classification, *Information Sciences*, 367 (5), 862-873.

14. Eroğul, U. (2009). *Sentiment analysis in Turkish*, Yüksek Lisans Tezi, Orta Doğu Teknik Üniversitesi, Ankara, 5-55.

15. Kaya, M., Fidan, G., and Toroslu, I. H. (2012, December). *Sentiment analysis of turkish political news.* In 2012 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology, 174-180.

16. Şimşek, M. U., and Özdemir, S. (2012, October). *Analysis of the relation between Turkish twitter messages and stock market index.* 6th International Conference on Application of Information and Communication Technologies, 1-4.

17. Aytekin, Ç. (2013). An opinion mining task in Turkish language: A model for assigning opinions in Turkish blogs to the polarities. *Journalism and Mass Communication*, 3 (3), 179-198.

18. Sevindi, B.İ., 2013, *Türkçe Metinlerde Denetimli ve Sözlük Tabanlı Duygu Analizi Yaklaşımlarının Karşılaştırılması*, Yüksek Lisans Tezi, Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara, 4-85.

19. Li, Y., Guzman, E., Tsiamoura, K., Schneider, F., and Bruegge, B. (2015, January). Automated Requirements Extraction for Scientific Software. *International Children's Continence Society*, 51 (2), 582-591.

20. Mu, Y., Wang, Y., and Guo, J. (2009, December). *Extracting software functional requirements from free text documents*. International Conference on Information and Multimedia Technology, 194-198.

21. Slankas, J., and Williams, L. (2013, May). *Automated extraction of non-functional requirements in available documentation.* In 2013 1st International Workshop on Natural Language Analysis in Software Engineering (NaturaLiSE), 9-16.

22. Cleland-Huang, J., Settimi, R., Zou, X., and Solc, P. (2006, September). *The detection and classification of non-functional requirements with application to early aspects*. In 14th IEEE International Requirements Engineering Conference (RE'06), 39-48.

23. Casamayor, A., Godoy, D., and Campo, M. (2010). Identification of non-functional requirements in textual specifications: A semi-supervised learning approach. *Information and Software Technology*, 52 (4), 436-445.

24. Frakes, W., and Succi, G. (2001). An Industrial Study of Reuse, Quality and Productivity, *Journal of Systems and Software*, 57 (2), 99-106.

25. Badareen, A., Selamat, M., Jabar,. Din, M., J, and Turaev, S. (2011). Reusability Software Component Life Cycle, *International Journal of Computers*, 5 (2), 191-200.

26. Rellermeyer, J. S., Alonso, G., and Roscoe, T. (2007, November). *R-OSGi: distributed applications through software modularization*. In ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing, 1-20.

27. Wang, Y. (2015). *Semantic Information Extraction for Software Requirements using Semantic Role Labeling*, IEEE International Conference on Progress in Informatics and Computing, 5-16.

28. Hamza, M., and Walker, R. (2015). *Recommending Features and Feature Relationships from Requirements Documents for Software Product Line*, 2015 4th International Workshop on Realizing AI Synergies in Software Engineering, 25-31.

29. Kahraman, E., İpek, T., İyidir, B., Bazlamaçcı, C., ve Bilgen, S. (2009). *Bileşen Tabanlı Yazılım Ürün Hattı Geliştirmeye Yönelik Alan Mühendisliği Çalışmaları*, Ulusal Yazılım Mühendisliği Sempozyumu'09, 283-286.

30. Doğan, S., ve Diri, B. (2010). Türkçe dokümanlar için N-gram tabanlı yeni bir sınıflandırma (Ng-ind): yazar, tür ve cinsiyet. *Türkiye Bilişim Vakfı Bilgisayar Bilimleri ve Mühendisliği Dergisi*, 3 (1), 11-19.

31. Berry, M. W. (2004). Survey of text mining. *Computing Reviews*, 45 (9), 548.

EKLER

EK-1. Doküman çözümleme ve ön işleme adımları

```
package project;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Set;

import org.apache.pdfbox.pdmodel.PDDocument;
import org.apache.pdfbox.util.PDFTextStripper;

import net.zemberek.tr.yapi.TurkiyeTurkcesi;
import net.zemberek.yapi.Kelime;
import zemberek.core.logging.Log;
import zemberek.morphology.TurkishMorphology;
import zemberek.normalization.TurkishSpellChecker;
import zemberek.tokenization.Token;
import zemberek.tokenization.TurkishTokenizer;
import zemberek.tokenization.Token.Type;

public class main {


    public static void main(String[] args) throws IOException {

        File[] yetenekDokumanlari;
        File onemsizlerDosyasi;
        File ozelKelimelerDosyasi;
        List <String>metin=new ArrayList<String>();
      List <String>yetenekler=new ArrayList<String>();
      List <String>tumKelimeArama=new ArrayList<String>();
      List <String>yetenekInput=new ArrayList<String>();
      List <String>yetenekKelimeler=new ArrayList<String>();
      List<String> eslesenYetenekDizisi = new ArrayList<String>();
      List <String>yetenekDizisiTumKelimeEslesenler=new ArrayList<String>();
      List <Kelime[]>ygoKelimeleriAsciiTolerans=new ArrayList<Kelime[]>();
      String input="",ygoText ="",st1="";
      List <String>onemsizKelimeler=new ArrayList<String>();
      List <String>ozelKelimeler=new ArrayList<String>();
      List <String>tamlamalar=new ArrayList<String>();
      HashMap<String,List<String>>sonuc=new HashMap<String,List<String>>();
      HashMap<String,List<String>>yetenekSonuc=new HashMap<String,List<String>>();
      HashMap<String,List<String>>hiyerarsikYetenekSonuc=new HashMap<String,List<String>>();

      PDDocument document = PDDocument.load(new File("C:\\7775.pdf"));
      if (!document.isEncrypted()) {
          PDFTextStripper stripper = new PDFTextStripper();
          ygoText = stripper.getText(document);
      }


      ZemberekSınıf zemberekSınıf=new ZemberekSınıf(new TurkiyeTurkcesi());

      TurkishTokenizer tokenizer = TurkishTokenizer.ALL;
      TurkishMorphology morphology = TurkishMorphology.createWithDefaults();
      TurkishSpellChecker spellChecker = new TurkishSpellChecker(morphology);
      StringBuilder output = new StringBuilder();
     for (Token token : tokenizer.tokenize(ygoText)) {
       String text = token.getText();
       if (analyzeToken(token) && !spellChecker.check(text)) {
         List<String> strings = spellChecker.suggestForWord(token.getText());
         if (!strings.isEmpty()) {
           String suggestion = strings.get(0);
           metin.add(text);
           Log.info("Correction: " + text + " -> " + suggestion);
           output.append(suggestion);
         } else {
           output.append(text);
           metin.add(text);
         }
       } else {
         output.append(text);
         metin.add(text);
       }
     }

     for(String mtn: metin) {
         if(zemberekSınıf.kelimeDenetle(mtn)){
             Kelime[] kelimeAsciiTolerans = zemberekSınıf.asciiCozumle(mtn);
             ygoKelimeleriAsciiTolerans.add(kelimeAsciiTolerans);
         }
     }
```

Şekil 1.1. Zemberek ile çözümlenen ygö dokümanındaki kelimelerin işlenmesi

EK-1. (devam) Doküman çözümleme ve ön işleme adımları

```
        onemsizlerDosyasi = dizindekiDosyayiGetir("onemsizkelimeler.txt","C:\\");

        BufferedReader br = new BufferedReader(new InputStreamReader(new FileInputStream(onemsizlerDosyasi), "UTF-8"));

        String st;
        while ((st = br.readLine()) != null) {
            onemsizKelimeler.add(st);
        }

        ozelKelimelerDosyasi = dizindekiDosyayiGetir("ozelkelimeler.txt","C:\\");

        BufferedReader br2 = new BufferedReader(new InputStreamReader(new FileInputStream(ozelKelimelerDosyasi), "UTF-8"));

        String st2;
        while ((st2 = br2.readLine()) != null) {
            ozelKelimeler.add(st2);
        }

        yetenekDokumanlari=dizindekiDosyalariGetir("C:\\yetenekler");

        for (int x = 0; x < yetenekDokumanlari.length; x++) {
            BufferedReader br1 = new BufferedReader(new InputStreamReader(new FileInputStream(yetenekDokumanlari[x]), "UTF-8"));

            while ((st1 = br1.readLine()) != null) {
                tumKelimeArama.add(st1);
            }
        }

        for(String klm:tumKelimeArama){
            if(ygoText.contains(klm)||ygoText.contentEquals(klm)){
                yetenekDizisiTumKelimeEslesenler.add(klm);
            }
        }
        System.out.println("Tum Kelime Arama icin Eslesme Sayisi:"+yetenekDizisiTumKelimeEslesenler.size());
        for(String eslsn:yetenekDizisiTumKelimeEslesenler){
            System.out.println("eslesen: "+eslsn);
        }
        System.out.println("----------------------------------------");
        ////////////////////////////////////////////////

        for (int x = 0; x < yetenekDokumanlari.length; x++) {
            BufferedReader br1 = new BufferedReader(new InputStreamReader(new FileInputStream(yetenekDokumanlari[x]), "UTF-8"));
            yetenekInput=new ArrayList<String>();
            yetenekKelimeler=new ArrayList<String>();
            while ((st1 = br1.readLine()) != null) {
                yetenekInput.add(st1);
            }

            yetenekler=new ArrayList<String>();
            tamlamalar=new ArrayList<String>();
            eslesenYetenekDizisi = new ArrayList<String>();
            yetenekKelimeler = yetenekDenetle(yetenekler, yetenekKelimeler, yetenekInput, zemberekSınıf, tokenizer,
                    spellChecker,ozelKelimeler,tamlamalar);

            yetenekSonuc=yetenekTespiti(yetenekKelimeler, eslesenYetenekDizisi, ygoKelimeleriAsciiTolerans,onemsizKelimeler,
                    yetenekDokumanlari[x],sonuc,hiyerarsikYetenekSonuc,ygoText,tamlamalar);
        }

        System.out.println("--------yetenekler----------");
        Set<String> keySet = yetenekSonuc.keySet();
        for(String key:keySet){
            System.out.println("key= "+ key);
            for(String value:yetenekSonuc.get(key)){
                System.out.print("value= "+ value +", ");
            }
            System.out.println();
        }
    }
```

Şekil 1.2. Dosyaların okunmasından sonra dokümandan yeteneklerin tespit edilmesi ve ekrana yazdırılması

EK-2. YGÖ dokümanından yetenek tespiti

```
private static HashMap<String, List<String>> yetenekTespiti(List<String> yetenekKelimeler, List<String> eslesenYetenekDizisi,
        List<Kelime[]> ygoKelimeleriAsciiTolerans,List <String>onemsizKelimeler, File yetenekDok,
        HashMap<String, List<String>> sonuc, HashMap<String, List<String>> hiyerarsikYetenekSonuc,
        String ygoText, List<String> tamlamalar) throws IOException {
    File elenecekKelimeler;
    List <String>elenecekler=new ArrayList<String>();
    for (Kelime[] std : ygoKelimeleriAsciiTolerans) {
        for (String ytnk : yetenekKelimeler) {
            if (onemsizKelimeler.contains(ytnk.toLowerCase()) == false) { //arasında onemli olmayan tek basina yetenek
                                                                          //belirtmeyen kelimeler elenir
                if ((std[0].icerikStr().toLowerCase().equals(ytnk.toLowerCase()))||(std[0].icerikStr().toLowerCase()
                        .contentEquals(ytnk.toLowerCase()))) {
                    if (eslesenYetenekDizisi.contains(ytnk) == false)
                        eslesenYetenekDizisi.add(ytnk);
                }
            }
        }
    }

    for (String tamlama : tamlamalar) {
        System.out.println("****************TAMLAMAAAA= "+tamlama);
        if (ygoText.toLowerCase().contains(tamlama.toLowerCase()) || ygoText.toLowerCase().contentEquals(tamlama.toLowerCase())) {
            if (eslesenYetenekDizisi.contains(tamlama) == false)
                eslesenYetenekDizisi.add(tamlama);
        }
    }

          if (dosyaDizindeVarMi(yetenekDok.getName(),"C:\\elenecekkelimeler\\")) {
              elenecekKelimeler = dizindekiDosyayiGetir(yetenekDok.getName(),"C:\\elenecekkelimeler\\");
              BufferedReader br3 = new BufferedReader(
                      new InputStreamReader(new FileInputStream(elenecekKelimeler), "UTF-8"));
              String st3;
              int say = 0;
              while ((st3 = br3.readLine()) != null) {
                  elenecekler.add(st3);
              }
              for (String value : eslesenYetenekDizisi) {
                  if (elenecekler.contains(value) == false) {
                      if (say == 0){
                          System.out.println("key= " + yetenekDok.getName());
                          hiyerarsikYetenekSonucOlustur(yetenekDok, hiyerarsikYetenekSonuc);
                      }
                      say++;
                      System.out.print("value= " + value + ", ");
                  }
              }
              System.out.println();
          } else {
              System.out.println("key= " + yetenekDok.getName());
              hiyerarsikYetenekSonucOlustur(yetenekDok, hiyerarsikYetenekSonuc);
              for (String value : eslesenYetenekDizisi) {
                  System.out.print("value= " + value + ", ");
              }
              System.out.println();
          }
      }
      return hiyerarsikYetenekSonuc;
  }
```

Şekil 2.1. Niteleyici kelimelerin ve kelime tamlamalarının dokümanda aranması

EK-3. Dosya işlemleri ve yeteneklerin ekrana yazılması

```
private static void hiyerarsikYetenekSonucOlustur(File yetenekDok,HashMap<String, List<String>> hiyerarsikYetenekSonuc) {
    String[] fileName = yetenekDok.getName().split("_");
    List<String> talent = new ArrayList<String>();
    for (int i = 0; i < fileName.length; i++) {
        if (fileName[i].contains(".txt")) {
            fileName[i].replace(".txt", "");
            if (hiyerarsikYetenekSonuc.keySet().contains(fileName[i]) == false)
                hiyerarsikYetenekSonuc.put(fileName[i], talent);
            else {
                List<String> yetenekSeti = hiyerarsikYetenekSonuc.get(fileName[i]);
                for (String tt : talent)
                    yetenekSeti.add(tt);
                hiyerarsikYetenekSonuc.remove(fileName[i]);
                hiyerarsikYetenekSonuc.put(fileName[i], yetenekSeti);
            }
        } else {
            talent.add(fileName[i]);
        }
    }
}

 public static boolean dosyaDizindeVarMi(final String fileName, final String folderPath)
 {
     final File folder = new File(folderPath);
     if (folder.exists()&& folder.isDirectory()){
        final File[] dosyaList = folder.listFiles();
        for (int i = 0; i < dosyaList.length; i++){
            if (dosyaList[i].getName().equalsIgnoreCase(fileName))
                return true;
        }
     }
     return false;
 }
public static File dizindekiDosyayiGetir(final String fileName, final String folderPath) {
    final File folder = new File(folderPath);
     if (folder.exists()&& folder.isDirectory()){
        final File[] dosyaList = folder.listFiles();
        for (int i = 0; i < dosyaList.length; i++){
            if (dosyaList[i].getName().equals(fileName))
                return dosyaList[i];
        }
     }
     return null;
}

public static File[] dizindekiDosyalariGetir(final String folderPath)
{
    final File folder = new File(folderPath);
    if (folder.exists()&& folder.isDirectory())
        return folder.listFiles();
    return null;
}

static boolean analyzeToken(Token token) {
    return token.getType() != Type.NewLine
        && token.getType() != Type.SpaceTab
        && token.getType() != Type.UnknownWord
        && token.getType() != Type.RomanNumeral
        && token.getType() != Type.Unknown;
  }
```

Şekil 3.1. Dosya dizin işlemleri ve yeteneklerin hiyerarşik şekilde ekrana yazılması

EK-4. Niteleyici kelimelerin işlenmesi ve tamlamaların oluşturulması

```
   private static List<String> yetenekDenetle(List<String> yetenekler, List<String> yetenekKelimeler, List<String> yetenekInput,
           ZemberekSınıf zemberekSınıf, TurkishTokenizer tokenizer, TurkishSpellChecker spellChecker,
           List<String> ozelKelimeler, List<String> tamlamalar) {
       StringBuilder output1 = new StringBuilder();
       List<String> yetenekInputTemp=new ArrayList<String>();
       List<String> yetenekKelimelerTemp=new ArrayList<String>();
       for(String ytnkinpt : yetenekInput){ //genel yeteneklerden satır silinmesin diye kopyalanıyor.
           yetenekInputTemp.add(ytnkinpt);
       }
       for (String ytnkinpt : yetenekInput) {
           for (Token token : tokenizer.tokenize(ytnkinpt)) {
               String text = token.getText();
'              System.out.println("text= "+text);
               if (analyzeToken(token) && !spellChecker.check(text)) {
                   List<String> strings = spellChecker.suggestForWord(token.getText());
                   if (!strings.isEmpty()) {
                       String suggestion = strings.get(0);
                       yetenekler.add(text);
                       output1.append(text);
                   } else {
                       output1.append(text);
                       yetenekler.add(text);
                   }
               } else {
                   output1.append(text);
                   yetenekler.add(text);
               }
           }
           String kelimeTamlamasi="";
   for (String ytnk : yetenekler) {
       if(ozelKelimeler.contains(ytnk)){ //"göndermec" gibi kelimeleri denetleyerek eleme yapmaması için ozel kelimeler ayriliyor
               yetenekKelimeler.add(ytnk);
               yetenekKelimelerTemp.add(ytnk);
               if (kelimeTamlamasi != "")
                   kelimeTamlamasi += " ";
               kelimeTamlamasi += ytnk;
               if(kelimeTamlamasi.equals(ytnk)==false)
                   yetenekKelimeler.add(kelimeTamlamasi);
       }
       if (zemberekSınıf.kelimeDenetle(ytnk)) {
           Kelime[] kelime = zemberekSınıf.kelimeCozumle(ytnk);

           for(Kelime k:kelime) {
               //çekim ekleri elenmiş olacak
               if((k.sonEk().toString().equals("ISIM_SAHIPLIK_O_I"))
                       ||(k.sonEk().toString().equals("ISIM_TAMLAMA_I"))||(k.sonEk().toString().equals("ISIM_BELIRTME_I"))
                       ||(k.sonEk().toString().equals("ISIM_SAHIPLIK_ONLAR_LERI"))){
                   // "Listesi" kelimesi için "liste" kok alinacak
                   boolean tamlamaOluscak=false;
                   if(yetenekKelimelerTemp.contains(k.kok().icerik())==false){
                       if ((k.kok().icerik().equals("kümes")==false)&&(k.kok().icerik().equals("ayn")==false)
                               &&k.kok().icerik().equals("payla")==false&&(k.kok().icerik().equals("pay")==false)
                               &&(k.kok().icerik().equals("ör")==false) //örüntü->ör
                               &&(k.kok().icerik().equals("kayıp")==false)&&(k.kok().icerik().equals("oyna")==false)
                               //oynatım->oyna  yayılım kaybı->yayılım kayıp olduğu için "oynatım" ve "kaybı" kelimeleri
                               //özel gruba eklenip diğerleri ihmal ediliyor
                               &&(k.kok().icerik().equals("reji")==false)&&(k.kok().icerik().equals("rejim")==false)
                               &&(k.kok().icerik().equals("söz")==false)) { //sözlük kelimesi tercih ediliyor söz kökü ihmal edilir
                           yetenekKelimelerTemp.add(k.kok().icerik());
                           yetenekKelimeler.add(k.kok().icerik());
                           if (kelimeTamlamasi != ""){
                               kelimeTamlamasi += " ";
                               tamlamaOluscak=true;
                           }
```

Şekil 4.1. Niteleyici kelimelerin eklerinden temizlenmesi ve kelime tamlamalarının oluşturulması

EK-4. (devam) Niteleyici kelimelerin işlenmesi ve tamlamaların oluşturulması

```
                        kelimeTamlamasi += k.kok().icerik();
                        if((tamlamaOluscak)&&(tamlamalar.contains(kelimeTamlamasi)==false))
                            tamlamalar.add(kelimeTamlamasi);
                        break;
                    }
                }
            }
            else{
                boolean tamlamaOluscak=false;
                if(yetenekKelimelerTemp.contains(ytnk)==false){
                    yetenekKelimelerTemp.add(ytnk);
                    yetenekKelimeler.add(ytnk); //isim kok ise dogrudan eklenecek
                    if ((kelimeTamlamasi != "")){
                        kelimeTamlamasi += " ";
                        tamlamaOluscak=true;
                    }
                    kelimeTamlamasi+=ytnk;
                    if((tamlamaOluscak)&&(tamlamalar.contains(kelimeTamlamasi)==false))
                        tamlamalar.add(kelimeTamlamasi);
                    break;
                }
            }
        }
        for(String input : yetenekInputTemp) {
            List<Token> array = tokenizer.tokenize(input);
            if(ytnk.equals(array.get(array.size()-1).content)){
                yetenekKelimeler.add(kelimeTamlamasi);
                yetenekInputTemp.remove(0);
                if(tamlamalar.contains(kelimeTamlamasi)==false)
                    tamlamalar.add(kelimeTamlamasi);
                kelimeTamlamasi="";
            }
            break;
        }
    }
      else{
          if(ytnk.equals(" ")==false){
              boolean tamlamaOluscak=false;
              if (yetenekKelimeler.contains(ytnk) == false) {
                  if (kelimeTamlamasi != ""){
                      kelimeTamlamasi += " ";
                      tamlamaOluscak=true;
                  }
                  kelimeTamlamasi += ytnk;
                  yetenekKelimeler.add(ytnk);
                  if((tamlamaOluscak)&&(tamlamalar.contains(kelimeTamlamasi)==false))
                      tamlamalar.add(kelimeTamlamasi);
              }
          }
      }
    }
    yetenekler=new ArrayList<String>();
    yetenekKelimelerTemp=new ArrayList<String>();
}
return yetenekKelimeler;
}
```

Şekil 4.2. YGÖ dokümanı içinde aratılacak kelime gruplarının oluşturulması

EK-5. Zemberek kütüphanesi sınıfı

```
package project;

import net.zemberek.bilgi.kokler.KokAdayiBulucu;
import net.zemberek.bilgi.ZemberekAyarlari;
import net.zemberek.bilgi.kokler.Sozluk;
import net.zemberek.islemler.AsciiDonusturucu;
import net.zemberek.islemler.HeceIslemleri;
import net.zemberek.islemler.KelimeKokFrekansKiyaslayici;
import net.zemberek.islemler.KelimeTabanliKokBulucu;
import net.zemberek.islemler.KelimeUretici;
import net.zemberek.islemler.KokBulucu;
import net.zemberek.islemler.cozumleme.*;
import net.zemberek.yapi.*;

import java.util.*;

public class ZemberekSınıf {

    private KelimeCozumleyici cozumleyici;
    private KelimeUretici kelimeUretici;
    private KelimeCozumleyici asciiToleransliCozumleyici;
    private OneriUretici oneriUretici;
    private AsciiDonusturucu asciiDonusturucu;
    private HeceIslemleri heceIslemleri;
    private Heceleyici heceleyici;
    private DilBilgisi dilBilgisi;
    private ZemberekAyarlari ayarlar;

    public ZemberekSınıf(DilAyarlari dilayarlari) {
        ayarlar = new ZemberekAyarlari(dilayarlari.locale().getLanguage());
        this.dilBilgisi = new TurkceDilBilgisi(dilayarlari);
        initialize();
    }

    private void initialize() {
        //Sozluk hazirla.
        Sozluk kokler = dilBilgisi.kokler();
        //Normal denetleyici-cozumleyici olusumu
        KokAdayiBulucu kokBulucu = kokler.getKokBulucuFactory().getKesinKokBulucu();
        cozumleyici = new StandartCozumleyici(
                kokBulucu,
                new KesinHDKiyaslayici(),
                dilBilgisi.alfabe(),
                dilBilgisi.ekler(),
                dilBilgisi.cozumlemeYardimcisi());

        // ASCII-Turkce donusturucu icin tukce toleransli cozumleyici olusumu.
        KokAdayiBulucu turkceToleransliKokBulucu = kokler.getKokBulucuFactory().getAsciiKokBulucu();
        asciiToleransliCozumleyici = new StandartCozumleyici(
                turkceToleransliKokBulucu,
                new AsciiToleransliHDKiyaslayici(),
                dilBilgisi.alfabe(),
                dilBilgisi.ekler(),
                dilBilgisi.cozumlemeYardimcisi());

        KokAdayiBulucu toleransliBulucu = kokler.getKokBulucuFactory().getToleransliKokBulucu(1);
        ToleransliCozumleyici toleransliCozumleyici = new ToleransliCozumleyici(
                toleransliBulucu,
                dilBilgisi.ekler(),
                dilBilgisi.alfabe(),
                dilBilgisi.cozumlemeYardimcisi());

        oneriUretici = new OneriUretici(
                dilBilgisi.cozumlemeYardimcisi(),
                cozumleyici,
                asciiToleransliCozumleyici,
                toleransliCozumleyici,ayarlar
        );
```

Şekil 5.1. Zemberek kütüphanesi fonksiyonları 1

EK-5. (devam) Zemberek kütüphanesi sınıfı

```
        asciiDonusturucu = new AsciiDonusturucu(dilBilgisi.alfabe());
        heceleyici = new Heceleyici() {

            @Override
            public List<String> hecele(HarfDizisi dizi) {
                return heceleyici.hecele(dizi);
            }
        };
        heceIslemleri = new HeceIslemleri(dilBilgisi.alfabe(), dilBilgisi.heceBulucu());

        kelimeUretici = new KelimeUretici(dilBilgisi.alfabe(), dilBilgisi.cozumlemeYardimcisi());
    }

    public boolean kelimeDenetle(String giris) {
        return cozumleyici.denetle(giris);
    }
    public String[] hecele(String giris) {
        return heceIslemleri.hecele(giris);
    }
    public Kelime[] kelimeCozumle(String giris) {
        return cozumleyici.cozumle(giris);
    }

    public KokBulucu kokBulucu() {
        return new KelimeTabanliKokBulucu(cozumleyici, dilBilgisi.alfabe());
    }

    public Kelime[] asciiCozumle(String giris) {
        Kelime[] sonuclar = asciiToleransliCozumleyici.cozumle(giris);
        Arrays.sort(sonuclar, new KelimeKokFrekansKiyaslayici());
        return sonuclar;
    }
     public String[] asciidenTurkceye(String giris) {
         Kelime[] kelimeler = asciiCozumle(giris);
         Set<String> olusumlar = new HashSet(kelimeler.length);
         for (Kelime kelime : kelimeler) {
             String olusum = kelime.icerik().toString();
             if (!olusumlar.contains(olusum))
                 olusumlar.add(olusum);
         }
         return olusumlar.toArray(new String[olusumlar.size()]);
     }
     public String[] oner(String giris) {
         return oneriUretici.oner(giris);
     }
     public String kelimeUret(Kok kok, List ekler) {
         return kelimeUretici.kelimeUret(kok, ekler);
     }
     /**
      * Istenilen kelimenin olasi String acilimlarini bulur.
      * Ornegin, "alayim" icin
      * "al-a-yim" ve "ala-yim" cozumleri String dizileri seklinde uretilir.
      */
     public List<List<String>> kelimeAyristir(String kelime) {
//         Set<List<String>> sonuclar = new HashSet();
         List<String[]>sonuclar=new ArrayList<String[]>();
         Kelime[] cozumler = cozumleyici.cozumle(kelime);
         for (Kelime kel : cozumler) {
             sonuclar.add(kelimeUretici.ayristir(kel));
         }
         return new ArrayList(sonuclar);
     }
```

Şekil 5.2. Zemberek kütüphanesi fonksiyonları 2

EK-6. Proje 1 YGÖ dokümanından tüm kelime arama çıktısı

```
eslesen: Nokta
eslesen: Polar
eslesen: Histogram
eslesen: Konum
eslesen: Yön
eslesen: Mesafe
eslesen: Yükseklik
eslesen: Bölge
eslesen: Hedef Listesi
eslesen: Alarm Listesi
eslesen: Teknik Listesi
eslesen: Platform Listesi
eslesen: Silah Listesi
eslesen: Platform Listesi
eslesen: Silah Listesi
eslesen: Radar Listesi
eslesen: Platform Listesi
eslesen: Silah Listesi
eslesen: Alarm Listesi
eslesen: Teknik Listesi
eslesen: Teknik Listesi
eslesen: Elektronik Taarruz
eslesen: ET Tarama Rejimi
eslesen: Radar Listesi
eslesen: Radar Listesi
eslesen: Kullanıcı Notu
eslesen: Yardım
eslesen: Tema
eslesen: Sistem Listesi
eslesen: Alarm Listesi
eslesen: Göndermeç Listesi
```

Şekil 6.1. Tüm kelime arama ile elde edilen çıktı

EK-7. Niteleyici kelimelerle oluşturulan tamlamalar

```
****************TAMLAMAAAA= Sistem liste
****************TAMLAMAAAA= Alarm liste
****************TAMLAMAAAA= Göndermeç liste
****************TAMLAMAAAA= Yayılım Kaybı analiz
****************TAMLAMAAAA= Arabakış Duyma
****************TAMLAMAAAA= Arabakış Duyma analiz
****************TAMLAMAAAA= Karıştırma Etkinlik
****************TAMLAMAAAA= Karıştırma Etkinlik analiz

****************TAMLAMAAAA= Kayıt oynatım
****************TAMLAMAAAA= Ses Çalma
****************TAMLAMAAAA= Durdur
****************TAMLAMAAAA= Oynat
****************TAMLAMAAAA= Duraklat
****************TAMLAMAAAA= Ses kayıt
****************TAMLAMAAAA= Ses kayıt listeleme
****************TAMLAMAAAA= İleri Sarma
****************TAMLAMAAAA= Geri Sarma
****************TAMLAMAAAA= Ses kayıt filtreleme

****************TAMLAMAAAA= Uçuşa yasak
****************TAMLAMAAAA= Uçuşa yasak bölge
****************TAMLAMAAAA= Uçuşa yasak bölge işlem
****************TAMLAMAAAA= Uçuş görev
****************TAMLAMAAAA= Uçuş görev dosya
****************TAMLAMAAAA= Uçuş görev dosya oluşturma
****************TAMLAMAAAA= Uçuş görev Kopyalama
****************TAMLAMAAAA= Uçuş görev kaydetme
****************TAMLAMAAAA= Uçuş görev yükleme
****************TAMLAMAAAA= Uçuş görev raporlama

****************TAMLAMAAAA= Radar EH
****************TAMLAMAAAA= Radar EH önerilen
****************TAMLAMAAAA= Radar EH önerilen uçuş
****************TAMLAMAAAA= Radar EH önerilen uçuş patika
****************TAMLAMAAAA= Radar EH önerilen uçuş patika gösterme
****************TAMLAMAAAA= Muhabere önerilen
****************TAMLAMAAAA= Muhabere önerilen uçuş
****************TAMLAMAAAA= Muhabere önerilen uçuş patika
****************TAMLAMAAAA= Muhabere önerilen uçuş patika gösterme

****************TAMLAMAAAA= Uçuş patika
****************TAMLAMAAAA= Uçuş patika doğrulama
****************TAMLAMAAAA= Uçuş patika doğrulama işlem
****************TAMLAMAAAA= Uçuşa yasak
****************TAMLAMAAAA= Uçuşa yasak bölge
****************TAMLAMAAAA= Uçuşa yasak bölge kontrol
****************TAMLAMAAAA= Yer şekil
****************TAMLAMAAAA= Yer şekil kontrol
****************TAMLAMAAAA= Yakıt kontrol
```

Şekil 7.1. Niteleyici ifadelerle oluşturulan tamlamalar 1

EK-7. (devam) Niteleyici kelimelerle oluşturulan tamlamalar

```
****************TAMLAMAAAA= Temas rapor
****************TAMLAMAAAA= Analiz rapor
****************TAMLAMAAAA= Dar Bant
****************TAMLAMAAAA= Dar Bant Tarama
****************TAMLAMAAAA= Dar Bant Tarama Rejimi
****************TAMLAMAAAA= Elektronik Destek
****************TAMLAMAAAA= Elektronik Destek Dar
****************TAMLAMAAAA= Elektronik Destek Dar Bant
****************TAMLAMAAAA= Elektronik Destek Dar Bant Tarama
****************TAMLAMAAAA= Elektronik Destek Dar Bant Tarama Rejimi
****************TAMLAMAAAA= Geniş Bant
****************TAMLAMAAAA= Geniş Bant Tarama
****************TAMLAMAAAA= Geniş Bant Tarama Rejimi
****************TAMLAMAAAA= Elektronik Destek Geniş
****************TAMLAMAAAA= Elektronik Destek Geniş Bant
****************TAMLAMAAAA= Elektronik Destek Geniş Bant Tarama
****************TAMLAMAAAA= Elektronik Destek Geniş Bant Tarama Rejimi
****************TAMLAMAAAA= İnceleme Tarama
****************TAMLAMAAAA= İnceleme Tarama Rejimi
****************TAMLAMAAAA= Elektronik Destek İnceleme
****************TAMLAMAAAA= Elektronik Destek İnceleme Tarama
****************TAMLAMAAAA= Elektronik Destek İnceleme Tarama Rejimi
****************TAMLAMAAAA= Elektronik Taarruz
****************TAMLAMAAAA= Elektronik Taarruz Tarama
****************TAMLAMAAAA= Elektronik Taarruz Tarama Rejimi
****************TAMLAMAAAA= Destek Tespit
****************TAMLAMAAAA= Destek Tespit Tarama
****************TAMLAMAAAA= Destek Tespit Tarama Rejimi
****************TAMLAMAAAA= Dar Bant
```

Şekil 7.2. Niteleyici ifadelerle oluşturulan tamlamalar 2

EK-7. (devam) Niteleyici kelimelerle oluşturulan tamlamalar

```
*****************TAMLAMAAAA= Binary Dosya
*****************TAMLAMAAAA= Binary Dosya Oluşturma
*****************TAMLAMAAAA= Text Dosya
*****************TAMLAMAAAA= Text Dosya Oluşturma
*****************TAMLAMAAAA= Radar Kopyalama
*****************TAMLAMAAAA= teknik Kopyalama
*****************TAMLAMAAAA= Platform Kopyalama
*****************TAMLAMAAAA= Silah Kopyalama
*****************TAMLAMAAAA= Radar Ekleme
*****************TAMLAMAAAA= Teknik Ekleme
*****************TAMLAMAAAA= Platform Ekleme
*****************TAMLAMAAAA= Silah Ekleme
*****************TAMLAMAAAA= Mod Kopyalama
*****************TAMLAMAAAA= Mod Ekleme
*****************TAMLAMAAAA= Binary Dosya
*****************TAMLAMAAAA= Binary Dosya Oluşturma
*****************TAMLAMAAAA= Text Dosya
*****************TAMLAMAAAA= Text Dosya Oluşturma
*****************TAMLAMAAAA= Radar Kopyalama
*****************TAMLAMAAAA= teknik Kopyalama
*****************TAMLAMAAAA= Platform Kopyalama
*****************TAMLAMAAAA= Silah Kopyalama
*****************TAMLAMAAAA= Radar Ekleme
*****************TAMLAMAAAA= Teknik Ekleme
*****************TAMLAMAAAA= Platform Ekleme
*****************TAMLAMAAAA= Silah Ekleme
*****************TAMLAMAAAA= Radar EH
*****************TAMLAMAAAA= Radar EH yerleşim
*****************TAMLAMAAAA= Radar EH yerleşim analiz
*****************TAMLAMAAAA= Radar EH yerleşim analiz yapma
*****************TAMLAMAAAA= Plan Kaydetme
*****************TAMLAMAAAA= Plan Yükleme
*****************TAMLAMAAAA= Plan Kopyalama
*****************TAMLAMAAAA= Plan Versiyonlama
*****************TAMLAMAAAA= Plan Raporlama
*****************TAMLAMAAAA= Çoklu Sistem
*****************TAMLAMAAAA= Çoklu Sistem plan
*****************TAMLAMAAAA= birden çok
*****************TAMLAMAAAA= birden çok görev
*****************TAMLAMAAAA= birden çok görev oluşturma
*****************TAMLAMAAAA= Aynı tipteki
*****************TAMLAMAAAA= Aynı tipteki sistemlerin
*****************TAMLAMAAAA= Aynı tipteki sistemlerin birlikte
*****************TAMLAMAAAA= Aynı tipteki sistemlerin birlikte plan
*****************TAMLAMAAAA= Farklı tipteki
*****************TAMLAMAAAA= Farklı tipteki sistemlerin
*****************TAMLAMAAAA= Farklı tipteki sistemlerin birlikte
*****************TAMLAMAAAA= Farklı tipteki sistemlerin birlikte plan
```

Şekil 7.3. Niteleyici ifadelerle oluşturulan tamlamalar 3

EK-8. Proje 1 yetenek çıktısı

```
--------yetenekler-----------
key= nitelik.txt
value= kutuphane, value= plan, value= senaryo,

key= plan.txt
value= muhabereeh, value= frekansatlamaliyayin,
value= muhabereeh, value= sabityayin,
value= radared,
value= radareh, value= radared,

key= senaryo.txt
value= radareh, value= kendinikoruma,
value= radareh, value= radared,
value= radareh, value= radaret,
value= radareh, value= taramarejimi,

key= genel.txt
value= grafik, value= harita,
value= listeyetenekleri, value= reform,
value= serbestanaliz,

key= kutuphane.txt
value= dısarayuz, value= radareh,
value= radareh, value= radaret,
value= radaret,
```

Şekil 8.1. Proje 1 ana ve alt yetenekler listesi

EK-9. Proje 1 yetenekler ve anahtar kelimeleri

```
key= grafik_genel.txt
value= Nokta, value= Grafik,
key= listeyetenekleri_genel.txt
value= dışa, value= içe, value= arama,
key= reform_genel.txt
value= Sürüm, value= şifre, value= Yardım, value= Tema,
value= Dil, value= Terimler, value= Hakkında,
key= serbestanaliz_genel.txt
value= Kaybı, value= Alarm, value= Arabakış Duyma,
value= Arabakış Duyma analiz, value= Karıştırma Etkinlik,
value= Göndermeç liste, value= Yayılım Kaybı analiz,
key= harita_genel.txt
value= Konum, value= Mesafe, value= Yön, value= Bölge, value= Yükseklik
-----------------------------------
key= muhabereeh_frekansatlamaliyayin_plan.txt
value= Almaç, value= Frekans Atlamalı,
key= muhabereeh_sabityayin_plan.txt
value= Almaç, value= Sabit Tehdit,
key= plan_nitelik.txt
value= Plan, value= Plan Kopyalama, value= Plan Raporlama,
key= radared_plan.txt
value= Platform, value= Silah,
key= radareh_radared_plan.txt
value= Platform, value= Silah, value= Platform liste, value= Silah list

key= grafik_genel.txt
value= Nokta, value= Grafik,
key= listeyetenekleri_genel.txt
value= dışa, value= içe, value= arama,
key= reform_genel.txt
value= Sürüm, value= şifre, value= Yardım, value= Tema,
value= Dil, value= Terimler, value= Hakkında,
key= serbestanaliz_genel.txt
value= Kaybı, value= Alarm,
value= Arabakış Duyma analiz,
value= Göndermeç liste, value= Yayılım Kaybı analiz,
key= harita_genel.txt
value= Konum, value= Mesafe, value= Yön, value= Bölge, value= Yükseklik,
-----------------------------------
key= muhabereeh_frekansatlamaliyayin_plan.txt
value= Almaç, value= Frekans Atlamalı,
key= muhabereeh_sabityayin_plan.txt
value= Almaç, value= Sabit Tehdit,
key= plan_nitelik.txt
value= Plan, value= Plan Kopyalama, value= Plan Raporlama,
key= radared_plan.txt
value= Platform, value= Silah,
key= radareh_radared_plan.txt
value= Platform, value= Silah, value= Platform liste, value= Silah liste,
```

Şekil 9.1. Proje 1 yetenekler ve dokümanda geçen anahtar kelimeler

EK-10. Proje 2 tüm kelime arama ile elde edilen çıktı

```
eslesen: Nokta
eslesen: Polar
eslesen: Histogram
eslesen: Konum
eslesen: Yön
eslesen: Yükseklik
eslesen: Bölge
eslesen: Hedef Listesi
eslesen: Alarm Listesi
eslesen: Çıkarılan Bant Listesi
eslesen: Platform Listesi
eslesen: Silah Listesi
eslesen: Platform Listesi
eslesen: Silah Listesi
eslesen: Radar Listesi
eslesen: Platform Listesi
eslesen: Silah Listesi
eslesen: Alarm Listesi
eslesen: Tarama Rejimi
eslesen: Dar Bant Histogram
eslesen: Radar Listesi
eslesen: Radar Listesi
eslesen: Operatör Notu
eslesen: Kullanıcı Notu
eslesen: Yardım
eslesen: Tema
eslesen: Alarm Listesi
```

Şekil 10.1. YGÖ dokümanından tüm kelime arama çıktısı

EK-11. Niteleyici kelimelerle oluşturulan tamlamalar

```
*****************TAMLAMAAAA= Aldatma liste
*****************TAMLAMAAAA= Kestirme liste
*****************TAMLAMAAAA= Kaydetme liste
*****************TAMLAMAAAA= Almaç atama
*****************TAMLAMAAAA= Göndermeç Atama
*****************TAMLAMAAAA= Konum Atama

*****************TAMLAMAAAA= Aldatma liste
*****************TAMLAMAAAA= Kestirme liste
*****************TAMLAMAAAA= Kaydetme liste
*****************TAMLAMAAAA= Almaç atama
*****************TAMLAMAAAA= Göndermeç Atama
*****************TAMLAMAAAA= Konum Atama

*****************TAMLAMAAAA= Uçuş patika
*****************TAMLAMAAAA= Uçuş patika gönderme
*****************TAMLAMAAAA= Muhabere EH
*****************TAMLAMAAAA= Muhabere EH uçuş
*****************TAMLAMAAAA= Muhabere EH uçuş patika
*****************TAMLAMAAAA= Muhabere EH uçuş patika gönderme

*****************TAMLAMAAAA= Kütüphane Sorgu
*****************TAMLAMAAAA= Kütüphane Kayıt
*****************TAMLAMAAAA= Kütüphane Kayıt Seçim
*****************TAMLAMAAAA= Tekli yayın
*****************TAMLAMAAAA= Tekli yayın ekleme
*****************TAMLAMAAAA= Çoklu yayın
*****************TAMLAMAAAA= Çoklu yayın ekleme

*****************TAMLAMAAAA= Liste eleman
*****************TAMLAMAAAA= Liste eleman raporlama
*****************TAMLAMAAAA= Liste eleman dışa
*****************TAMLAMAAAA= Liste eleman dışa aktarma
*****************TAMLAMAAAA= Liste eleman içe
*****************TAMLAMAAAA= Liste eleman içe aktarma
*****************TAMLAMAAAA= Liste eleman kopyalama
*****************TAMLAMAAAA= Liste eleman versiyonlama
*****************TAMLAMAAAA= Liste eleman kilitleme
*****************TAMLAMAAAA= Liste eleman sadece
*****************TAMLAMAAAA= Liste eleman sadece okunabilirlik
*****************TAMLAMAAAA= Liste eleman salt
*****************TAMLAMAAAA= Liste eleman salt okunabilirlik
*****************TAMLAMAAAA= Liste eleman arama
*****************TAMLAMAAAA= Liste eleman sayfalama
*****************TAMLAMAAAA= Liste eleman sıralama
*****************TAMLAMAAAA= Liste eleman filtreleme
*****************TAMLAMAAAA= Liste eleman ekleme
*****************TAMLAMAAAA= Liste eleman silme
*****************TAMLAMAAAA= Liste eleman güncelleme
```

Şekil 11.1. Niteleyici ifadelerle oluşturulan tamlamalar 1

EK-11. (devam) Niteleyici kelimelerle oluşturulan tamlamalar

```
****************TAMLAMAAAA= Muhabere önerilen
****************TAMLAMAAAA= Muhabere önerilen uçuş
****************TAMLAMAAAA= Muhabere önerilen uçuş patika
****************TAMLAMAAAA= Muhabere önerilen uçuş patika oluşturma
****************TAMLAMAAAA= Sabit Hedef
****************TAMLAMAAAA= Sabit Hedef ekleme
****************TAMLAMAAAA= Frekans atlamalı
****************TAMLAMAAAA= Frekans atlamalı Hedef
****************TAMLAMAAAA= Frekans atlamalı Hedef ekleme
****************TAMLAMAAAA= Hedef ekleme
****************TAMLAMAAAA= Muhabere EH
****************TAMLAMAAAA= Muhabere EH Yerleşim
****************TAMLAMAAAA= Muhabere EH Yerleşim analiz
****************TAMLAMAAAA= Muhabere EH Yerleşim analiz Yapma
****************TAMLAMAAAA= Almaç Göndermeç ilişki
****************TAMLAMAAAA= Baraj Karıştırma
****************TAMLAMAAAA= Baraj Karıştırma liste
****************TAMLAMAAAA= Reaktif Karıştırma
****************TAMLAMAAAA= Reaktif Karıştırma liste
****************TAMLAMAAAA= Hedef liste
****************TAMLAMAAAA= Alarm liste
****************TAMLAMAAAA= Ses Sayısal
****************TAMLAMAAAA= Ses Sayısal Kayıt
****************TAMLAMAAAA= Ses Sayısal Kayıt liste
****************TAMLAMAAAA= ses kayıt
****************TAMLAMAAAA= sayısal kayıt

****************TAMLAMAAAA= Çıkarılan Bant
****************TAMLAMAAAA= Çıkarılan Bant liste
****************TAMLAMAAAA= Muhabere Koruma
****************TAMLAMAAAA= Muhabere Koruma liste
****************TAMLAMAAAA= Drisa Dönme
****************TAMLAMAAAA= Drisa Dönme tehdit
****************TAMLAMAAAA= Drisa Dönme tehdit liste
```

Şekil 11.2. Niteleyici ifadelerle oluşturulan tamlamalar 2

EK-11. (devam) Niteleyici kelimelerle oluşturulan tamlamalar

```
*****************TAMLAMAAAA= Binary Dosya
*****************TAMLAMAAAA= Binary Dosya Oluşturma
*****************TAMLAMAAAA= Text Dosya
*****************TAMLAMAAAA= Text Dosya Oluşturma
*****************TAMLAMAAAA= Radar Kopyalama
*****************TAMLAMAAAA= Mod Kopyalama
*****************TAMLAMAAAA= teknik Kopyalama
*****************TAMLAMAAAA= Platform Kopyalama
*****************TAMLAMAAAA= Silah Kopyalama
*****************TAMLAMAAAA= Radar Ekleme
*****************TAMLAMAAAA= Mod Ekleme
*****************TAMLAMAAAA= teknik Ekleme
*****************TAMLAMAAAA= Platform Ekleme
*****************TAMLAMAAAA= Silah Ekleme

*****************TAMLAMAAAA= Log kayıt
*****************TAMLAMAAAA= Log kayıt Gösterme
*****************TAMLAMAAAA= Log kayıt Silme
*****************TAMLAMAAAA= Log kayıt Kaydetme
*****************TAMLAMAAAA= Lisanslama
*****************TAMLAMAAAA= Yetkilendirme
*****************TAMLAMAAAA= Kullanıcı şifre
*****************TAMLAMAAAA= Oturum Açma
*****************TAMLAMAAAA= Hakkında
*****************TAMLAMAAAA= Operatör not
*****************TAMLAMAAAA= Kullanıcı not
*****************TAMLAMAAAA= Sürüm bilgi
*****************TAMLAMAAAA= Çoklu Dil
*****************TAMLAMAAAA= Çoklu Dil destek
*****************TAMLAMAAAA= Birim Çevirme
*****************TAMLAMAAAA= Terimler sözlük
****************TAMLAMAAAA= Radar EH
****************TAMLAMAAAA= Radar EH yerleşim
****************TAMLAMAAAA= Radar EH yerleşim analiz
****************TAMLAMAAAA= Radar EH yerleşim analiz yapma
****************TAMLAMAAAA= Plan Kaydetme
****************TAMLAMAAAA= Plan Yükleme
****************TAMLAMAAAA= Plan Kopyalama
****************TAMLAMAAAA= Plan Versiyonlama
****************TAMLAMAAAA= Plan Raporlama
****************TAMLAMAAAA= Çoklu Sistem
****************TAMLAMAAAA= Çoklu Sistem plan
****************TAMLAMAAAA= birden çok
****************TAMLAMAAAA= birden çok görev
****************TAMLAMAAAA= birden çok görev oluşturma
****************TAMLAMAAAA= Aynı tipteki
****************TAMLAMAAAA= Aynı tipteki sistemlerin
****************TAMLAMAAAA= Aynı tipteki sistemlerin birlikte
****************TAMLAMAAAA= Aynı tipteki sistemlerin birlikte plan
****************TAMLAMAAAA= Farklı tipteki
****************TAMLAMAAAA= Farklı tipteki sistemlerin
****************TAMLAMAAAA= Farklı tipteki sistemlerin birlikte
****************TAMLAMAAAA= Farklı tipteki sistemlerin birlikte plan
*****************TAMLAMAAAA= Uçuşa yasak
*****************TAMLAMAAAA= Uçuşa yasak bölge
*****************TAMLAMAAAA= Uçuşa yasak bölge işlem
*****************TAMLAMAAAA= Uçuş görev
*****************TAMLAMAAAA= Uçuş görev dosya
*****************TAMLAMAAAA= Uçuş görev dosya oluşturma
*****************TAMLAMAAAA= Uçuş görev Kopyalama
*****************TAMLAMAAAA= Uçuş görev kaydetme
*****************TAMLAMAAAA= Uçuş görev yükleme
*****************TAMLAMAAAA= Uçuş görev raporlama
```

Şekil 11.3. Niteleyici ifadelerle oluşturulan tamlamalar 3

EK-12. Proje 2 yetenek çıktısı

```
--------yetenekler-----------
key= nitelik.txt
value= kutuphane, value= senaryo, value= ucusplanlama,

key= senaryo.txt
value= muhabereeh,
value= radared,
value= radareh, value= radared,
value= radareh, value= taramarejimi,

key= ucusplanlama.txt
value= ucuspatikasibelirleme,
value= ucuspatikasidogrulama,

key= genel.txt
value= dissistem, value= grafik,
value= harita, value= listeyetenekleri,
value= reform,

key= kutuphane.txt
value= radared,
value= radareh, value= radared,
```

Şekil 12.1. Proje 2 ana ve alt yetenekler listesi

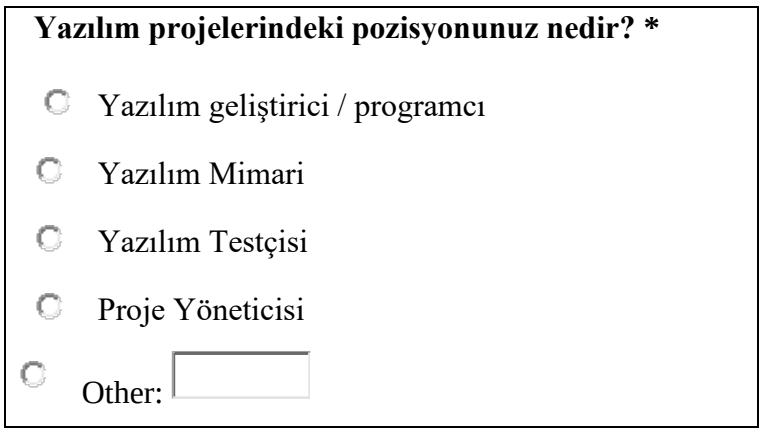
EK-13. Proje 2 yetenekler ve anahtar kelimeleri

```
key= dissistem_genel.txt
value= Uçuş, value= Uçuş patika,
key= listeyetenekleri_genel.txt
value= dışa, value= içe,
key= grafik_genel.txt
value= Nokta, value= Grafik,
key= harita_genel.txt
value= Konum, value= Bölge, value= Yön, value= Yükseklik,
key= reform_genel.txt
value= Sürüm, value= şifre, value= Yardım, value= Tema,
value= Dil, value= Terimler, value= Hakkında,
-----------------------------------------------
key= muhabereeh_senaryo.txt
value= uçuş, value= ses, value= Muhabere EH,
value= Çıkarılan Bant,
value= Çıkarılan Bant liste, value= Hedef liste,
value= Alarm liste,
key= radared_senaryo.txt
value= Platform, value= Silah,
key= senaryo_nitelik.txt
value= senaryo, value= Görev Veri,
key= radareh_radared_senaryo.txt
value= Platform, value= Silah, value= Alarm, value= Radar liste,
value= Platform liste, value= Silah liste, value= Alarm liste,
key= radareh_taramarejimi_senaryo.txt
value= Rejimi, value= Tespit,
value= Dar Bant, value=    Dar Bant Tarama,
value= Dar Bant Tarama Rejimi, value= Tarama,
value= Tarama Rejimi,
-----------------------------------------------
key= radareh_radared_kutuphane.txt
value= Platform, value= Silah, value= Füze,
value= Platform liste, value= Silah liste,
key= kutuphane_nitelik.txt
value= Kütüphane, value= Kütüphane Kopyalama,
key= radared_kutuphane.txt
value= Platform. value= Silah.
-----------------------------------------------
key= ucuspatikasibelirleme_ucusplanlama.txt
value= Uçuş Patika,
key= ucuspatikasidogrulama_ucusplanlama.txt
value= Uçuş, value= Uçuş patika,
key= ucusplanlama_nitelik.txt
value= Uçuş, value= Uçuş görev,
```

Şekil 13.1. Proje 2 yetenekler ve dokümanda geçen anahtar kelimeler

EK-14. Anket çalışması

Gerçekleştirilen Sistematik Literatür Taraması sonucunda Görev Planlama Yazılımının yeniden kullanılabilirliğini değerlendirmek ve Ürün Hattını oluşturabilmek için kullanılan en belirleyici yazılım faktörlerinin bağlaşım (İng. Coupling), uyumluluk (İng.Cohesion) ve karmaşıklık (İng. Complexity) olduğu sonucuna varılmıştır. Ortak yazılım ürün hattı ailesinden gelen OSGi çerçevesinde Java programlama dili kullanılarak geliştirilmiş yazılımların Yazılım Gereksinim Özellikleri dokümanları incelenmiştir. Yeniden kullanılabilecek nitelikte olan bileşenler bilir kişilerce kontrol edilerek tespit edilmiştir. Bu süreçte Hedef-Soru-Metrik yaklaşımı kullanılmıştır. Bir Görev Planlama yazılımında yer alabilecek tüm yetenekler incelenmiş ve yetenek setleri yazılım geliştirici ekip ile birlikte oluşturulmuştur. Dokümanlardan ayrıştırılan ifadeler, kelime havuzunda aranmış ve eşleşenler tespit edilmiştir. Havuzdaki kelimelerin hangi yeteneği temsil ettiği önceden belirtilmiş olduğu için eşleşen kelimelerden yetenek tespiti yapılabilmiştir. Böylece gereksinim dokümanında yer alan isteklerin, Görev Planlama yazılımındaki hangi yeteneğe karşılık geldiği tespit edilebilmiştir. Tespit edilen ana yeteneklerin alt yeteneklerle bağlantıları incelenerek ürün hattı oluşturulmuştur. Bu anket çalışması, OSGi çerçevesi, Java program dili kullanılarak geliştirilen yazılımların yeniden kullanılabilirliğini arttırmak ve gereksinim dokümanlarından otomatik yetenek çıkarımını sağlayabilmek için önerilen algoritmanın kullanılabilirliğini (İng. Usability) ve kullanılabilirlikteki kolaylığını (İng. Ease of Use) araştırmayı amaçlamaktadır.


Yazılım projelerindeki pozisyonunuz nedir? *

- Yazılım geliştirici / programcı
- Yazılım Mimari
- Yazılım Testçisi
- Proje Yöneticisi
- Other:

Şekil 14.1. Anket soruları 1

EK-14. (devam) Anket çalışması

Yazılım geliştirirken OSGi çerçevesini ne sıklıkta kullanıyorsunuz? *

- Her zaman
- Bazen
- Nadir
- Hiçbir zaman
- Other:

Yazılımın geliştirmenin hangi fazında yeniden kullanım yapıyorsunuz?*

- Gereksinim Belirleme Fazı
- Tasarım Fazı
- Uygulama/ Kodlama Fazı
- Test ve Entegrasyon Fazı
- Bakım Fazı
- Other:

Yazılım geliştirirken başvurduğunuz yeniden kullanım teknikleri nelerdir? *

- Tasarım Örüntüleri (İng. Desing Patterns)
- Çerçeveler (İng. Frameworks)
- API ve Kütüphaneler (API & Libraries)
- Üretici Programlama (İng. Generative Programming)
- Model Güdümlü Geliştirme (İng. Model Driven Development)
- Other:

Şekil 14.2. Anket soruları 2

EK-14. (devam) Anket çalışması

Yazılım geliştirirken yeniden kullanım tekniklerine hangi sıklıkta başvuruyorsunuz? *

- Her zaman
- Bazen
- Nadiren
- Hiçbir zaman
- Other:

Yeniden kullanılabilirliği arttırma ve ürün hattı oluşturma amaçlı otomatik yetenek çıkarımı yönteminin uygulanabilirliği nedir? *

- Uygulanabilirliği yüksek
- Uygulanabilirliği düşük
- Uygulanamaz
- Other:

Yeniden kullanılabilirliği arttırma ve ürün hattı oluşturma amaçlı otomatik yetenek çıkarımı yöntemini uygulamanın kolaylık derecesi nedir? *

- Çok kolay
- Kolay
- Zor
- Other:

Yeniden kullanılabilirliği arttırma ve ürün hattı oluşturma amaçlı otomatik yetenek çıkarımı yöntemini uygulamayı düşünür müsünüz? *

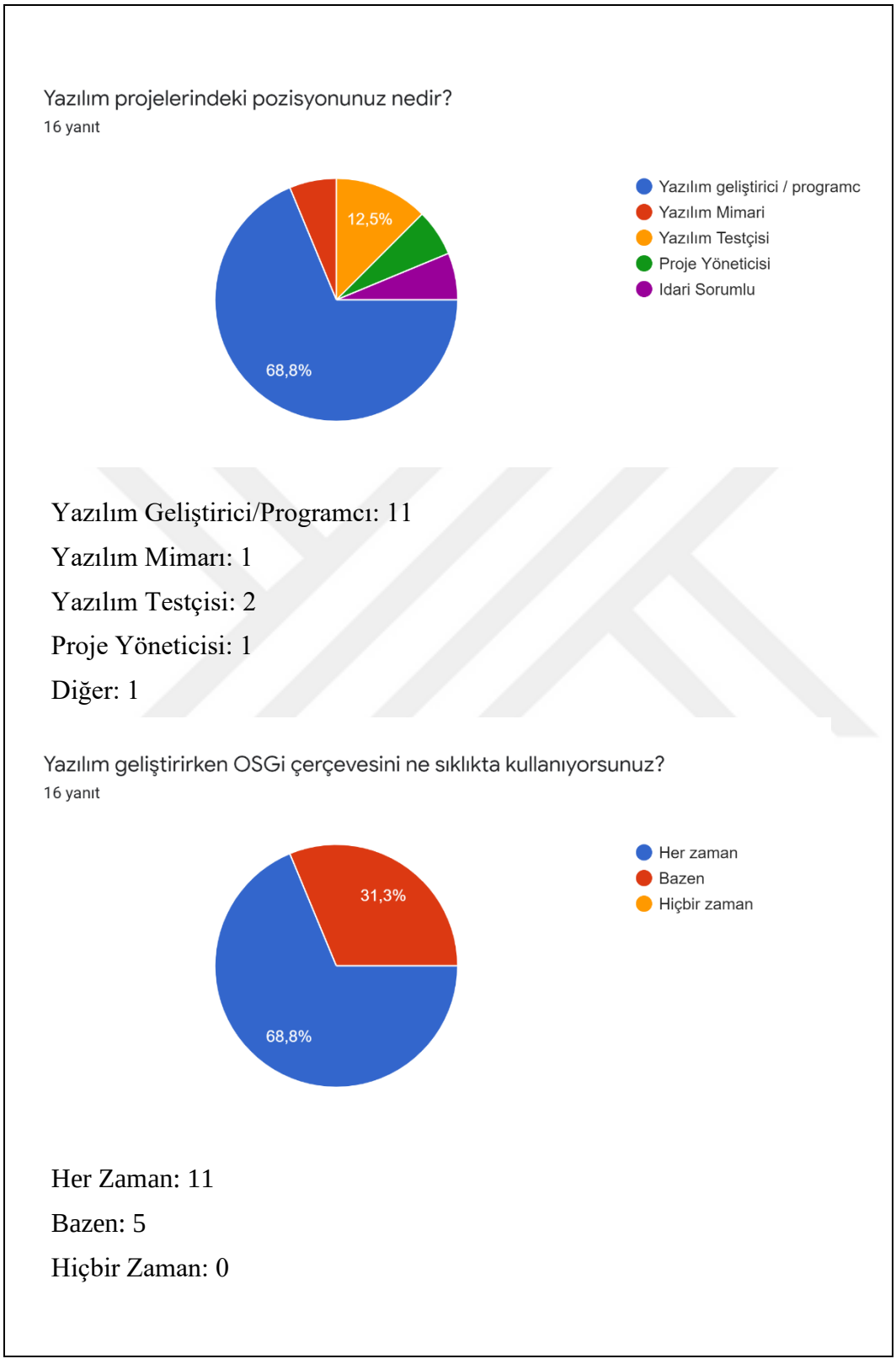
- Evet
- Hayır

Cevabınız hayırsa öngördüğünüz engeller nelerdir?

Submit

Şekil 14.3. Anket soruları 3

EK-15. Anket sonuçları



Şekil 15.1. Anket cevapları 1

EK-15. (devam) Anket sonuçları

Her Zaman: 12

Bazen: 4

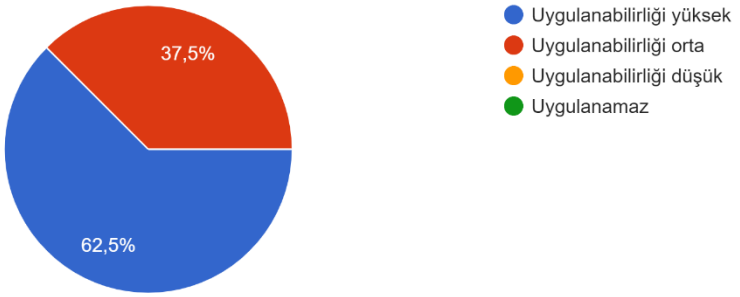
Hiçbir Zaman: 0

Şekil 15.2. Anket cevapları 2

EK-15. (devam) Anket sonuçları

Yeniden kullanılabilirliği arttırma ve ürün hattı oluşturma amaçlı otomatik yetenek çıkarımı yönteminin uygulanabilirliği nedir?

16 yanıt



Uygulanabilirliği Yüksek: 10

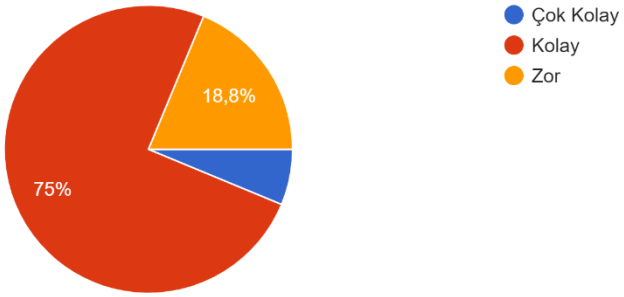
Uygulanabilirliği Orta: 6

Uygulanabilirliği Düşük: 0

Uygulanamaz: 0

Yeniden kullanılabilirliği arttırma ve ürün hattı oluşturma amaçlı otomatik yetenek çıkarımı yöntemini uygulamanın kolaylık derecesi nedir?

16 yanıt



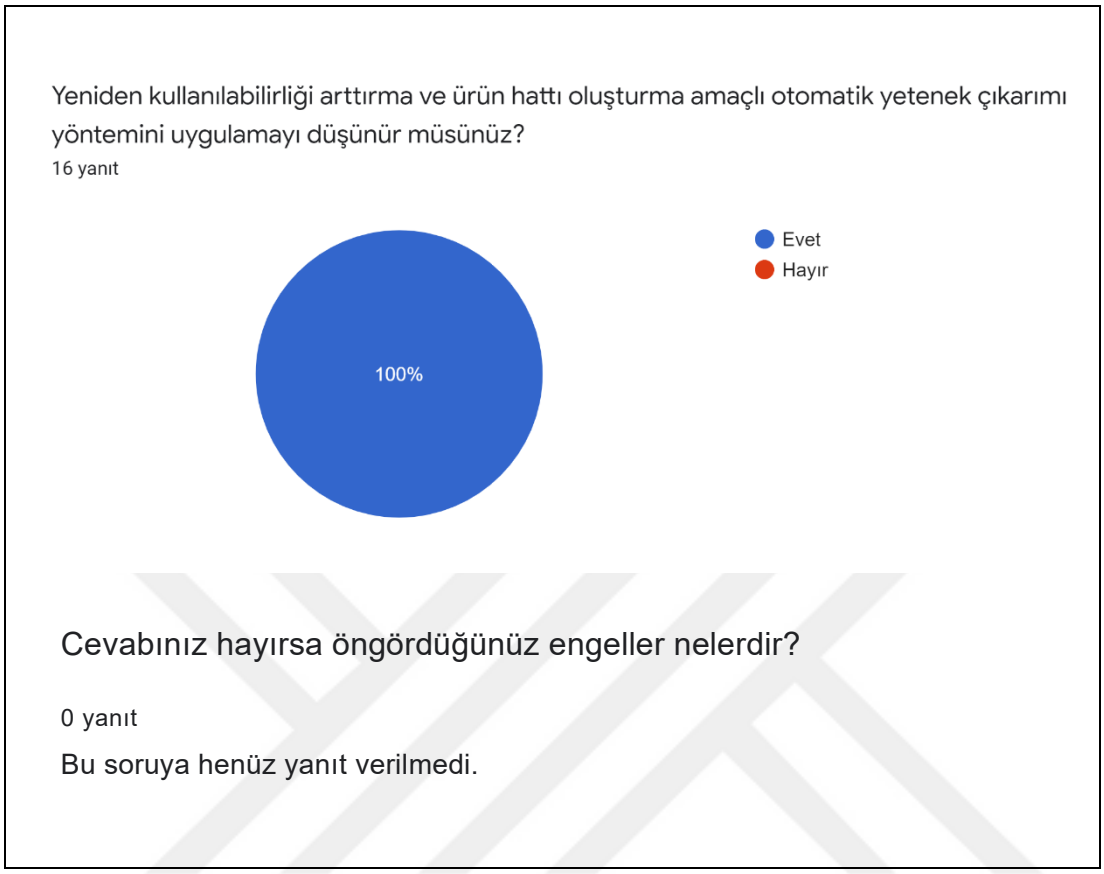
Çok Kolay: 1

Kolay: 12

Zor: 3

Şekil 15.3. Anket cevapları 3

EK-15. (devam) Anket sonuçları



Şekil 15.4. Anket cevapları 4

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı : BIYIKLI, Melis
Uyruğu : T.C.
Doğum tarihi ve yeri: 23.04.1995, Ankara
Medeni hali : Bekar
Telefon : 0 (534) 745 08 70
e-mail : byklmelis@gmail.com.tr

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Yüksek lisans	Gazi Üniversitesi / Bilgisayar Mühendisliği	Devam Ediyor
Lisans	Gazi Üniversitesi / Bilgisayar Mühendisliği	2017
Lise	Çankaya Lisesi	2013

İş Deneyimi

Yıl	Yer	Görev
2017-Halen	Aselsan A.Ş.	Mühendis

Yabancı Dil

İngilizce

Yayınlar

Bıyıklı M. (2020). *Radar ve elektronik harp projelerinde "Görev Planlama" yazılımları için yazılım ürün hattı mimarisi kapsamında yeniden kullanılabilirlik yöntemlerinin uygulanması*. 2nd International Eurasian Conference on Science.

Hobiler

Yüzme, Gitar

GAZİ GELECEKTİR...