

**İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ**

**METİN İŞLEME: SORU SORAN BİR  
SİSTEM TASARIMI**

**YÜKSEK LİSANS TEZİ  
Zeki MOCAN**

**Anabilim Dalı : BİLGİSAYAR MÜHENDİSLİĞİ**

**Programı : BİLGİSAYAR MÜHENDİSLİĞİ**

**OCAK 2005**

**METİN İŞLEME: SORU SORAN BİR  
SİSTEM TASARIMI**

**YÜKSEK LİSANS TEZİ  
Zeki MOCAN  
504031533**

**Tezin Enstitüye Verildiği Tarih : 27 Aralık 2004  
Tezin Savunulduğu Tarih : 25 Ocak 2005**

**Tez Danışmanı: Doç. Dr. Coşkun SÖNMEZ**

**Jüri Üyeleri: Prof. Dr. Eşref ADALI  
Prof. Dr. Tamer ÖLMEZ**

**OCAK 2005**



## ÖNSÖZ

Tezin konusu, metin işleme ve anlamaya dayalı soru soran bir sistem tasarımı üzerine kurulmuştur. Bu tür sistemler geçmişten bu yana sürekli gelişim göstermişler ve nihayet bugünkü duruma gelmişlerdir. Ancak bugün gelinen nokta dahi henüz bu alanda aşılması gereken daha birçok problem olduğunu göstermektedir. Bugün bile sistemlerin hata oranı yeterli derecede düşürülememiştir.

Soru soran sistemler ya da türevleri olan soru yanıtlayan sistemler, doğal dil işleme konusunun kapsamı altında incelenmektedir. Bu nedenle konuya geçmeden önce, doğal dil işleme sistemlerinde ortak olarak kullanılan bir takım teknikler incelenmektedir. Metin işleme ve anlamaya dayalı sistemler, birden ortaya çıkmamıştır. Bu nedenle araştırmada, bu tür sistemlerin doğmasına neden olan faktörler ve bu sistemlerin gelişim süreci incelenerek, öne çıkan bazı projeler irdelenmektedir.

Tezin konusuna yönelik geliştirilen uygulamada, bilgisayara girilmiş olan metinle ilgili, kullanıcıya çeşitli sorular yöneltilmektedir. Buradaki amaç; ilköğretim düzeyindeki öğrencilerin okuduklarını anlama düzeylerinin belirlenmesi ve bilgisayarlarla iletişimlerinin sağlanarak geliştirilmesidir. Bu sayede öğrencilerin bilgisayarlara olan bakış açısı da gelişecektir.

Bu tez çalışmam süresince her türlü anlamda yardımlarını esirgemeyen hocam Sayın Doç. Dr. Coşkun SÖNMEZ'e ve manevi desteklerini benden esirgemeyen aileme çok teşekkür ederim.

Ocak 2005

Zeki MOCAN

## İÇİNDEKİLER

	<b><u>Sayfa No</u></b>
<b>KISALTMALAR</b>	<b>vi</b>
<b>TABLO LİSTESİ</b>	<b>vii</b>
<b>ŞEKİL LİSTESİ</b>	<b>viii</b>
<b>ÖZET</b>	<b>ix</b>
<b>SUMMARY</b>	<b>xi</b>
<b>1. GİRİŞ</b>	<b>1</b>
<b>2. DOĞAL DİL İŞLEME-DDİ (NATURAL LANGUAGE PROCESSING)</b>	<b>4</b>
2.1. Biçim Bilimsel (Morfolojik) Analiz	7
2.2. Kelime Türünün Belirlenmesi (Part Of Speech- POS Operation)	9
2.3. Sözdizim (Sintaks) Analizi	10
2.4. Anlamsal (Semantik) Analiz	11
2.4.1. Önerme mantığı	11
2.4.2. Kavram çizgesi	11
2.4.3. Anlamsal ağ	11
2.5. Söylem (Discourse) Analizi	13
2.5.1. Söylem segmentasyonu (bölümlenmesi)	13
2.5.1.1. Stack kullanarak söylem segmentasyonu yapmak	13
2.6. Makine Çevirisi	14
2.6.1. Transfer-tabanlı makine çevirisi	14
2.6.2. Interlingua-tabanlı makine çevirisi	15
2.7. Doğal Dil İşlemede Geline Nokta	15
2.8. Türkçe’de Doğal Dil İşleme Alanına Giren Genel Konu Başlıkları	16
2.9. Türkiye’de Doğal Dil İşleme Üzerine Yapılan ve Yapılmakta Olan Projeler	17
2.9.1. Tamamlanmış projeler	17
2.9.2. Henüz tamamlanmış projeler	18
2.10. DDİ İle İlgili Bazı Önemli Akademik Yayınlar ve Konferanslar	18
<b>3. METİN İŞLEME VE ANLAMA TEMELLİ BAZI SİSTEMLER</b>	<b>20</b>
3.1. Soru Yanıtlama ve Sorma Sistemlerinin Tarihsel Gelişimi	20
3.2. LILOG (Linguistics and Logic) Projesi	22
3.2.1. LILOG sistemi ne yapar?	23
3.2.2. LILOG sisteminin yapamadıkları	24
3.3. Bir cevap çıkarım sistemi: ExtrAns	24
3.3.1. Geri çekimli arama stratejisi	25
3.3.2. ExtrAns sisteminde geliştirilmesi gereken noktalar	26

3.4. LOLITA (Large-scale, Object-based, Linguistic Interactor, Translator, and Analyser) Sistemi	26
3.4.1. LOLITA sisteminin mimarisi	27
3.4.2. LOLITA sisteminin oluşumu	28
<b>4. METİN İŞLEME ve ANLAMA TEMELLİ ARAŞTIRMALARIN UYGULAMA ALANLARI ve GELECEĞİ</b>	<b>29</b>
4.1. Uygulama Alanları	29
4.2. Geleceğe Yönelik Kullanım Alanları	31
<b>5. UYGULAMAYA YÖNELİK TEMEL BİLGİLER</b>	<b>33</b>
5.1. Türkçe Dilbilgisi Temel Kuralları	33
5.1.1. Türkçe’de kök ve ek ilişkisi	33
5.1.2. Çekim eklerinin kök ya da gövdeye bitişmesi	33
5.1.2.1. Fiil çekim ekleri	34
5.1.2.2. İsim çekim ekleri	34
5.2. Prolog	34
5.2.1. AMZI! Prolog	35
5.2.1.1. Mimarisi	35
5.2.1.2. Logic server engine ve LSAPI	35
5.2.1.3. Delphi bileşeni	36
5.2.1.4. Amzi! Prolog’un desteklediği ortamlar	36
5.3. Dll ( Dynamic Link Library ) Dosyaları	36
5.3.1. ISAPI/NSAPI genel yapısı	37
5.3.1.1. Delphi webbroker ve webmodule teknolojileri	37
5.3.1.2. Neden ISAPI teknolojisi?	38
<b>6. SORU SORAN SİSTEMİN GERÇEKLEŞTİRİLMESİ</b>	<b>39</b>
6.1. Temel Adımlar	39
6.1.1. Kelime Ayrıştırıcı (Kelimeparser)	39
6.1.2. CumleAyrıştırıcı (Cumleparser)	39
6.1.3. Öge Ayrıştırıcı (Ogeparser)	40
6.1.4. Prolog önermesine çevirme	40
6.1.5. Soru üretme	41
6.1.6. Yanıt işleme	41
6.1.7. Doğruluk kontrolü	41
6.2. Kelimelerin Kök ve Eklerine Ayrılması	41
6.2.1. Ek ve köklerine ayırma modülü	41
6.2.1.1. Olası köklerin bulunması	42
6.2.1.2. Çekim eklerinin bulunması	42
6.2.2. Analiz modülü	44
6.2.2.1. Fiilimsi_eki_analizi modülü	45

6.2.2.2. İsim_çekim_ekleri_analizi modülü	45
6.2.2.3. Fiil_çekim_ekleri_analizi modülü	47
6.2.3. Kelimenin cümle içerisindeki durumunun incelenmesi	47
6.2.4. Tamlamalara ait uygulama örnekleri	48
6.2.4.1. Sıfat tamlamaları	48
6.2.4.2. İsim tamlamaları	48
6.3. Cümlelerin Temel ve Yan Cümleciklere Ayrılması	49
6.3.1. Yapı bakımından cümleler	49
6.3.1.1. Basit cümle	50
6.3.1.2. Birleşik cümle	50
6.3.1.3. Sıralı cümleler	51
6.3.1.4. Bağlı cümleler	52
6.3.2. Cümle Ayrıştırıcı modülün işleyişi	52
6.4. Basit Cümlelerin Öğelerine Ayrılması	56
6.4.1. Cümlelerin öğeleri	56
6.4.1.1. Yüklem	57
6.4.1.2. Özne	57
6.4.1.3. Nesne	58
6.4.1.4. Dolaylı tümleş	58
6.4.1.5. Zarf tümleşci	59
6.4.2. Basit bir cümleyi öğelerine ayıran modülün çalışma adımları	59
6.5. Basit Cümlelerin ve Bağlantılarının Prolog Formatına Dönüştürülmesi	60
6.6. Soru Modülü	61
6.6.1. Özneye dayalı soru oluşturma modülü	62
6.6.2. Nesneye dayalı soru oluşturma modülü	62
6.6.3. Dolaylı tümleşce dayalı soru oluşturma modülü	63
6.6.4. Zarf tümleşcine dayalı soru oluşturma modülü	63
6.7. Yanıt Kontrol ve Doğrulama Modülü	64
<b>7. ARAYÜZ TASARIMI VE EKRAN GÖRÜNTÜLERİ</b>	<b>66</b>
7.1. Örnek Uygulama I	66
7.2. Örnek Uygulama II	71
<b>8. SİSTEM BAŞARISININ ÖLÇÜLMESİ</b>	<b>74</b>
<b>9. SONUÇLAR VE TARTIŞMA</b>	<b>76</b>
<b>KAYNAKLAR</b>	<b>79</b>
<b>ÖZGEÇMİŞ</b>	<b>81</b>

## KISALTMALAR

<b>FYE</b>	: Fiil Yapım Eki
<b>IYE</b>	: İsim Yapım Eki
<b>ytk</b>	: Yeterlilik Kipi
<b>ol</b>	: Olumsuzluk Eki
<b>SFE</b>	: Sıfat Fiilimsi Eki
<b>Ç</b>	: Çoğul Eki
<b>ys</b>	: Yardımcı Ses
<b>ieç1</b>	: İyelik Eki 1. Çoğul Şahıs
<b>ay</b>	: Ayrılma Hali (-den hali)
<b>bul</b>	: Bulunma Hali (-de hali)
<b>tm</b>	: Tamlayan Eki (-in eki)
<b>ku</b>	: Koruyucu Ünsüz
<b>ef_dgz</b>	: Ek Fiil Görülen Geçmiş Zaman Eki
<b>şet1</b>	: 1. Tekil Şahıs Eki
<b>yön</b>	: Yönelme Hali (-e hali)
<b>ef_gez</b>	: Ek Fiil Geniş Zaman Eki
<b>kh</b>	: Kaynaştırma Hali
<b>gez</b>	: Geniş Zaman
<b>dgz</b>	: Görülen (di'li) Geçmiş Zaman Eki
<b>mgz</b>	: Öğrenilen (miş'li) Geçmiş Zaman Eki
<b>şz</b>	: Şimdiki Zaman Eki
<b>ZFE</b>	: Zarf Fiilimsi Eki
<b>seç1</b>	: 1. Çoğul Şahıs Eki
<b>iet3</b>	: 3. Tekil İyelik Eki
<b>IFE</b>	: İsim Fiilimsi Eki
<b>(yön) Hal Eki</b>	: İsmi -e Hali
<b>(ayr) Hal Eki</b>	: İsmi -den Hali
<b>(bul) Hal Eki</b>	: İsmi -de Hali
<b>(bln) Hal Eki</b>	: İsmi -i Hali
<b>Pekiştirme bağ.</b>	: bile, de, hem de, dahi, üstelik, hatta, ayrıca
<b>Karşıtlık bağ.</b>	: ama, fakat, lakin, yalnız, ancak, ne var ki, ne yazık ki
<b>Gerekçe bağ.</b>	: çünkü, madem, zira, yoksa, nasıl ki, değil mi ki



## TABLO LİSTESİ

	<u>Sayfa No</u>
<b>Tablo 6.1.</b> Cumle Ayrıştırıcı Modülün İşleyişi .....	53
<b>Tablo 6.2.</b> Öğelerine Ayırma Kuralları.....	59
<b>Tablo 8.1.</b> Sistemin Basit Cümleleri Öğelerine Ayırma Başarısı.....	74
<b>Tablo 8.2.</b> Sistemin Soru Üretme Başarısı.....	75

## ŞEKİL LİSTESİ

	<b>Sayfa No</b>
<b>Şekil 2.1</b> : Genel Bir Doğal Dil İşleme Sistemi.....	4
<b>Şekil 2.2</b> : Genel bir doğal dil işleme sistemine ait bileşenlerin ardışık düzen gösterimi.....	4
<b>Şekil 2.3</b> : Genel bir doğal dil işleme sistemindeki katılımcı sürecin görünüşü.....	6
<b>Şekil 2.4</b> : Morfemlerin Yapısı.....	7
<b>Şekil 2.5</b> : Türkçe’de Eklemeli Morfoloji.....	8
<b>Şekil 2.6</b> : Türkçe’de Türetmeli Morfoloji.....	8
<b>Şekil 2.7</b> : Türkçe’de İsim Eklerinin Morfotaktik Sırası.....	8
<b>Şekil 2.8</b> : Türkçe’de Fiil Eklerinin Morfotaktik Sırası.....	8
<b>Şekil 2.9</b> : Papağanlara Dair Anlamsal Bir Ağ Örneği.....	12
<b>Şekil 2.10</b> : Tek Parça Akışıyla 500 Parçanın İşlenmesi.....	12
<b>Şekil 2.11</b> : Stack Kullanarak Söylem Segmantasyonu.....	14
<b>Şekil 2.12</b> : Transfer-tabanlı Makine Çevirisi.....	15
<b>Şekil 2.13</b> : Interlingua-tabanlı Makine Çevirisi.....	15
<b>Şekil 3.1</b> : SHARDLU’nun Çalışma Esnasındaki Orjinal Ekran Görüntüsü.	22
<b>Şekil 3.2</b> : ExtrAns Sisteminin Sorgulama Ekranı.....	25
<b>Şekil 3.3</b> : LOLITA Sisteminin Blok Diyagramı.....	27
<b>Şekil 6.1</b> : Sistem Akış Diyagramı.....	40
<b>Şekil 7.1</b> : Giriş Ekranı.....	70
<b>Şekil 7.2</b> : Metnin Ek ve Köklerine Ayrıldığı ve Herbir Bileşik Cümlelerin Basit Cümle Yapısına Dönüştürüldüğü Ekran.....	67
<b>Şekil 7.3</b> : Girilen metnin Öğelerine Ayrılığı Ekran.....	68
<b>Şekil 7.4</b> : Metinle İlgili Nesne Sorularının Çıkarıldığı Ekran.....	69
<b>Şekil 7.5</b> : Dolaylı Tümeleç Sorularının Çıkarıldığı Ekran.....	70
<b>Şekil 7.6</b> : Metnin internet üzerinden çözümlendiği ekran.....	71
<b>Şekil 7.7</b> : Girilen Metnin Basit Cümleler Haline Getirilmesi.....	72
<b>Şekil 7.8</b> : Girilen Metnin Öğelerine Ayrılmış Şekli.....	72
<b>Şekil 7.9</b> : Kullanıcıya soru yöneltilen ve girilen yanıtın kontrol edildiği ekran.....	73

## **METİN İŞLEME: SORU SORAN BİR SİSTEM TASARIMI**

### **ÖZET**

Günümüzde metin işleme ve anlama tabanlı yapay zeka uygulamaları gün geçtikçe, artan bir şekilde rağbet görmeye başlamıştır. Bunun en temel nedeni, bilgiye olan ihtiyacın daha da artması, dolayısıyla da bilgiye daha hızlı erişme gereksinimin doğmasıdır.

Metin işleme ve anlama tabanlı sistemlerde en çok kullanılanlar, soru soran ve soru yanıtlayan sistemlerdir. Soru yanıtlayan sistemlerde bilgisayar, kendisine verilen metinle ilgili, kullanıcı sorularını doğal dilde yanıtlar. Bir nevi kullanıcı ile bilgisayar arasında tam bir etkilişimin sağlandığı sistemlerdir. Burada amaç; kullanıcının erişmek istediği bilgiye, kendi dilinde, adeta aradığını bir insana sorarmış gibi erişmesini sağlamak ve bilgiyi aramak için harcanan zamanı ortadan kaldırmaktır.

Soru soran sistemler ise, yeni gelişmekte olan sistemlerdir ve bu tez çalışmasının ana konusunu oluşturmaktadır. Bu sistemlerin en yaygın olarak kullanılabileceği alan, eğitim alanıdır. Şu an doğal dil işleme sistemlerinin geldiği noktaya bakıldığında, henüz ilköğretim seviyesindeki öğrencilere yönelik sistemler geliştirilebilmektedir. Soru soran sistemlerde bilgisayar, öğretmenin girdiği metinle ilgili öğrencilere soru sorabilmekte ve öğrencilerin okuduğunu anlama düzeyleri bu şekilde kontrol edilebilmektedir.

Bu tez çalışması, yapay zekanın bir alanı olan doğal dil işleme (NLP - Natural Language Processing) teknolojisi kapsamında gerçekleştirilmiştir. Daha önce yapılmış olan projelerden yararlanılmış ve Türkçe doğal dil işleme çalışmasının daha da geliştirilmesi hedeflenmiştir.

Bundan sonraki NLP projelerinde hazır olarak kullanılabilecek, internet üzerinde çalışabilen modüler NLP araçları ve dökümanlarının oluşturulması projenin ilk

adımını oluşturmıştır. Bu NLP araçları “kelime ayrıştırıcı”, “cümle ayrıştırıcı”, “öge ayrıştırıcı” ve “sözlük” ’ dır.

Kelime ayrıştırıcı modülde, girilen metindeki tüm kelimeler “kök-gövde-yapımeği-çekimeki” formatında ayrılmıştır. Cümle ayrıştırıcı modülde, yan cümleler içeren bileşik cümleler basit cümlelere ayrılarak, bu basit cümleler birbirleriyle ilişkilendirilmiştir. Öge ayrıştırıcı modülde, basit cümlelerin öğeleri bulunmuştur. Sözlük, diğer modüllerin kullandığı yaklaşık 50000 kelimelik bir sözlüktür ve sözcüklerin kullanım sıklıklarına göre türlerinin belirlenmiş olduğu bir veritabanı yapısındadır.

Bu çalışmanın ikinci adımında, girilen bir metindeki cümleler analiz edilerek, kullanıcıya cümlelerle ilgili sorular sorulmuş ve kullanıcının verdiği cevap incelenerek doğruluğu kontrol edilmiştir.

Girilen metin öncelikle eklerine ayrılmıştır. Daha sonra yan cümleler içeren karmaşık cümleler, basit cümleciklere ayrılmış ve aralarındaki bağlantılar tespit edilmiştir. Son olarak basit cümleler öğelerine ayrılarak Prolog önermelerine çevrilmiş ve Prolog veritabanına eklenmiştir.

Kullanıcının bu sorulara verdiği yanıtlar benzer şekilde öğelerine ayrılarak, doğruluğu kontrol edilmiştir. Kullanıcının yanlış yanıt vermesi durumunda, doğru yanıt kullanıcıya sunulmuştur.

Tez çalışmasının amacı, yeni gelişmekte olan doğal dilde metin işleme ve anlama tabanlı soru soran sistemlere yönelik Türkçe dahilinde bir çalışma yapmaktır. Bu çalışma sırasında, Türkçe’ye yönelik doğal dil işleme çalışmalarının henüz yeterli olmadığı ve eldeki kaynakların çok da tatmin edici olmadığı saptanmıştır. Ayrıca dünyada gelinen noktaya bakıldığında metin işleme çalışmaları her ne kadar yeterli düzeyde olsa da, metinsel anlama henüz tam olarak başarılamamıştır. Ancak bu çalışmaların yıllar gerektiren çalışmalar olduğu düşünülürse, geleceğe yönelik karamsar bir tablo çizmek son derece yanlış olacaktır.

## **TEXT PROCESSING: A QUESTION ASKING SYSTEM DESIGN**

### **SUMMARY**

These days, the applications about text processing and understanding have started attracting attention. The main reason of this is, the increasing need of information and the need of accessing to information more quickly.

The being used mostly within text processing and understanding systems are question asking and question answering systems. The main principle of question asking systems is that the computer answers the questions of a user about the text given in natural language form which means building an interactive system that provides a link between the user and computer. The aim here is, to provide an environment to user that makes possible the accessing any information much more quickly and accessing in his/her own natural language as if asking a question to a human being facing which also reduces the time wasting during searching of the information.

Question asking systems are newly developed systems and they are the main subject of this thesis. These systems can be used most widely in education area. That is possible to develop systems which are for just primary school students if looking at the place we are present at text processing and question asking systems. In these systems, the teacher inputs a text to computer, and then computer asks questions about the input text. In this way, the text understanding level of the student can be compared.

This work has been developed in the research area of Artificial Intelligence, Natural Language Processing(NLP). Earlier projects have been used as a source. It is benefited from the earlier projects and aimed to widen the horizons of language processing studies.

The first step of the project is creating web-based moduleer NLP tools which can run on the Internet, and NLP documentaries. These NLP tools are “kelimeparser”, “cumleparser”, “ogeparser” and “dictionary” respectively.

In the kelimeparser module, all words in the text are parsed according to the format of root-conjugation affix. In the cumleparser module, by separating complex sentences, which includes more than one verb, into simple sentences, these simple sentences are related to each other in respect of connection between them. In the ogeparser module, all components of sentences are defined. Dictionary, which has about 50000 words, is used by the other modulees and it has a database structure that keeps also the part-of-speech of the words which means the type of words.

In the second step of this work, by analyzing sentences in input text, questions are represented to user and answers are examined.

Once input text entered, all words are parsed into its roots and affixes. Then complex sentences are parsed into simple sentences and the relationships between them are defined. Finally, simple sentences are broken into their components and then they are translated to Prolog clauses and added to Prolog database.

Similarly, users’ answers are translated to Prolog form, and then examined for validity. In the case of detecting the wrong answers, the right ones are displayed to the user.

The major reason of this work is to make a research about the Turkish text processing and understanding systems which are also newly developping in the world in other languages. During these work it has been realized that the works about the natural language processing in Turkish is not enough and the resources we have is not satisfied. Although, text processing works can be seen satisfied enough in the world, text understanding systems are really far from the point that it must be in. But if it were thought that these kind of works need so much time to be completed fully, to think pessimistic about the future of this science would be wrong also.

## 1. GİRİŞ

Bilgisayar teknolojilerindeki gelişmelerin ilk yıllarından itibaren insanlar, kendi aralarında kurdukları iletişimin bir benzerini bilgisayarlar ile sağlamayı istemişlerdir. İnsan tarafından yapıldığında zeka olarak adlandırılan davranışların (akıllı davranışların) makina tarafından da yapılabilmesi hedeflenmiştir. Bunun için insan aklının nasıl çalıştığını gösteren bir kuram olan Yapay Zeka'dan yararlanılmıştır. Bu kuram ile çeşitli problemlerde optimal çözümü bulabilen bilgisayar yazılımları geliştirilmiştir. Ancak, bugün bile bu konuda istenilen düzeye gelinememiştir.

Yapay zeka'nın bir alanı olan doğal dil işleme, ana işlevi doğal bir dili çözümleme, anlama, yorumlama ve üretme olan bilgisayar sistemlerinin tasarımını ve gerçekleştirilmesini konu alan bir bilim ve mühendislik alanıdır. Doğal dil işlemenin uygulamalarında; makinaların doğal insan dillerini kullanabilmesi, yazılan metinleri anlayarak doğal dilde cevap verebilmesi, veritabanları ve robot sistemleri arasında arabirim oluşturma gibi konular üzerinde çalışılmaktadır. Dil tabanlı yapay zeka araştırmalarının amacı, anlamsal bulguları, sonuç üretecek birimin kullanabileceği hazır formlara dönüştürmektir.

Doğal dil işleme alanında birçok projeler yapılmıştır ve de yapılmaktadır. Tercüman programları, ses tanıma sistemleri (text-to-speech), görüntü sıralarının bilgisayar tarafından algılanarak doğal dil tanımlarına çevrilmesi bu alandaki çalışmalardandır. Bu çalışmalar içinde en önemlilerden biri de ELIZA projesidir. Bu proje ile insanların Yapay Zeka'ya olan ilgisi artmıştır. ELIZA projesinde yapay bir psikiyatrist yaratılmış ve insanların ruhsal sorunlarını giderme amaçlı bir sistem oluşturulmuştur. Bu çalışmalar bilgisayar bilimcilerini, dil bilimcilerini ve hatta psikoloji bilimcilerini biraraya getirmiştir. Daha sonra ELIZA projesinin tam bir yapay zeka ürünü olmadığı hakkında çeşitli eleştiriler gündeme gelmiştir. Ancak, ELIZA, insanların ilgisini yapay zekaya yönlendiren bir çalışma olduğundan tarihteki yerini almıştır.

Birçok dilde uygulanan doğal dil işleme tekniklerinin, dünyada en yaygın altıncı dil olan ve Asya ile Avrupa’da yoğun olarak konuşulan Türkçe’de de kullanılması araştırma alt yapısı oluşturması açısından yararlı olacaktır.

Türkçe, yapım ve çekim eklerini içeren sondan eklemeli bir dildir. Eklerin sonda yer alması sözcüklerin eklerine ve köklerine ayrılması işlemini zorlaştırmaktadır. Kelimelere eklenen ekler, kelimenin anlamını ve cümle içindeki görevini belirlemektedir. Bu çalışmada Türkçe metin analizi ve sorgulama yapabilen bir sistem geliştirilecektir.

Tezin ikinci bölümünde doğal dil işleme-ddi (natural language processing) kavramına genel bir giriş yapılmaktadır. Doğal dil işlemenin temel adımları olan, morfolojik (biçim bilimsel) analiz, kelime türünün belirlenmesi, sözdizim analizi, anlamsal analiz, söylem analizi ve makine çevirisi adımları anlatılmakta, doğal dil işlemede bugün gelinen nokta irdelenmekte ve Türkçe’de doğal dil işleme alanında yapılmış ve yapılmakta olan projelere değinilmekte, ddi alanında yararlanılabilecek akademik yayınlar ve konferanslar belirtilmektedir.

Tezin üçüncü bölümünde, tezin konusunu da içine alan, metin anlama, işleme tabanlı sistemler ile soru sorma ve yanıtlama sistemleri, bu sistemlerin tarihsel gelişim süreci ve bu sistemlerde öne çıkan LILOG, ExtrAns, LOLITA gibi örnekler ve çalışma prensipleri incelenmektedir.

Tezin dördüncü bölümünde bu sistemlerin uygulanma alanları ve insanlığa ne gibi katkı sağlayabilecekleri incelenmektedir. Bu bölümde ayrıca bu sistemlerin geleceği ve bu sistemler üzerinde yapılabilecek geliştirme çalışmaları öngörülmektedir.

Beşinci bölümde Türkçe dilbilgisinde kullanılan temel kurallara yer verilmekte, yapay zeka alanında en çok kullanılan dillerden biri olan PROLOG dilinin genel yapısı, PROLOG dilinin çeşitli versiyonlarından biri olan Amzi! Prolog’un mimarisi ve sahip olduğu “Logic Server Engine-Mantıksal Sunucu Motoru” incelenmektedir. Bu motorun Delphi için hazırlanmış olan bileşeni ve bu bileşenin Delphi içinden doğrudan kullanılması yine bu bölümde belirtilmiştir. Bu bölümde ayrıca DLL (Dynamic Link Library) dosyalarının genel yapısı, sağladığı avantajlar, ISAPI ve NSAPI mimarileri, Delphi’nin WebBroker ve WebModule teknolojileri ile neden tez kapsamında ISAPI teknolojisinin seçildiği anlatılmaktadır.



Altıncı bölümde projenin adımları ve genel olarak hangi modüllerden oluştuğu irdelenmektedir. Bu bölümde kelimelerin ek ve köklerine ayrılması, Türkçe'deki temel cümle yapıları ve cümlelerin temel ve yan cümlelere ayrılması, basit cümlelerin öğelerine ayrılması, basit cümlelerin ve birbiriyle bağlantılı basit cümlelerden oluşan birleşik cümlelerin bu bağlantılarıyla birlikte Prolog yapısına dönüştürülmesi işlemi, ve son olarak sistemin soru modülünün tasarımı ve işleyişi anlatılmaktadır. Bu modül, kullanıcıya metinle ilgili sorular yönelten modüldür.

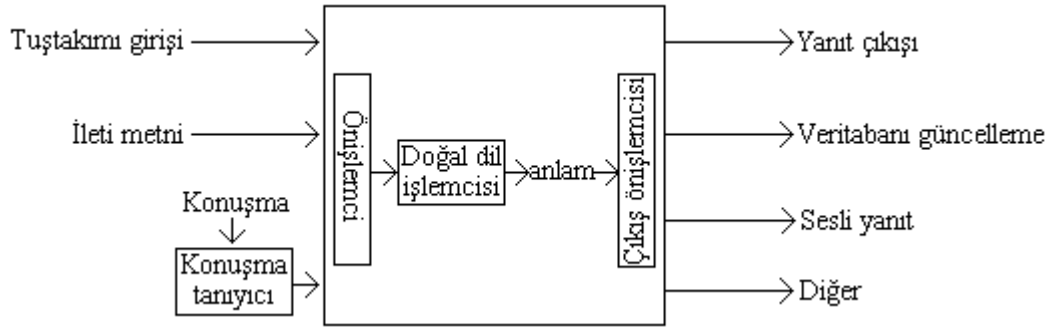
Yedinci bölümde programın ekran görüntüleri sistemin çalışmasına dair bir örnek olması açısından gösterilmekte ve iki uygulamaya yer verilmektedir.

Sekizinci bölümde ise sistemin ne kadar doğru çalıştığını saptamak üzere sistem başarı yüzdesinin belirlenmesini amaçlayan bir takım sına sonuçlarına değinilmektedir.

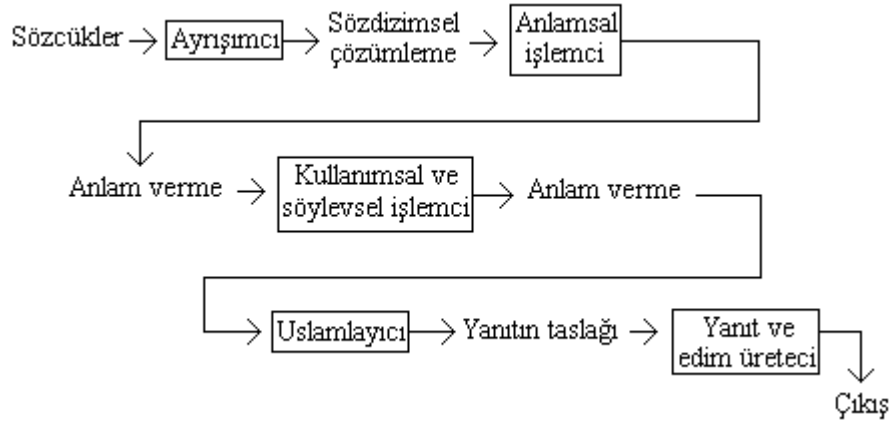
Son olarak dokuzuncu bölümde tezin konusuna dair sonuç ve tartışma bölümü yer almaktadır. Bu çalışmada geleceğe yönelik yapılabilecek iyileştirmeler ve geliştirmeler üzerine bir tartışma da yine bu bölümde belirtilmektedir.

## 2. DOĞAL DİL İŞLEME-DDİ (NATURAL LANGUAGE PROCESSING-NLP)

Şekil 2.1'de genel bir doğal dil işleme sistemi giriş ve çıkış değişkenleriyle birlikte görülmektedir. Şekil 2.2'de ise Şekil 2.1'deki blok şemanın içinde tipik olarak nelerin bulunduğu gösterilmektedir. Şekil 2.2'deki bloklardan her biri doğal dil işlemeyi oluşturan işlemlerden birini temsil eder [8].



Şekil 2.1. Genel Bir Doğal Dil İşleme Sistemi [8]



Şekil 2.2. Genel bir doğal dil işleme sistemine ait bileşenlerin ardışık düzen gösterimi [8]

Çoğu doğal dil işleme sisteminde başkalaşımsal analizler yapan, sözlükleri ayıran, sözcükleri sınıflandıran ve onları birbirleriyle karşılaştıran bir “önişlemci” bulunur. Yukarıdaki işlemlerin gerçekleştirilme sırası ve tekniği ile çıkışın biçimi sistemden sisteme değişir [8].

Problemi basitleştirme: Doğal dil işleme sistemlerinin tamamı Şekil 2.2'de gösterilen bileşenlerin tümüne birden sahip olacak diye bir kural yoktur. Bazı sistemlerin anlamı çıkarmak için çok fazla sözdizimsel bilgiye gereksinim duymayan ayrışımçıları bulunabilmektedir. Diğerleri sözdizimi ve anlam bilgisi kurallarını harmanlamışlardır. Bazı uygulamalarsa kullanımsal ve söyleysel işlemlerden birisine ya da her ikisine birden çok az ihtiyaç duyarlar [8].

Bir kısım sistem Şekil 2.2'deki bileşenlerin hemen hemen hepsini atarak, sözcükleri doğrudan -hiçbir düzeyde detaylı dilsel analizine ihtiyaç duymaksızın anlamlı bir yanıt üretmeye çalışan- bir uslamlayıcıya (bir uzman sistem olabilir) yöneltir. Yalnızca birkaç çıkışa izin verilen uygulamalardaysa uslamlayıcı ve yanıt üretici de kaldırılabilir [8].

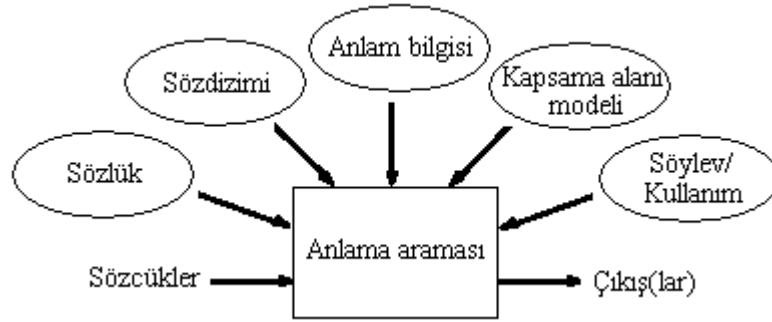
Yukarıda bazı sistemlerin doğal dil işleme paketindeki araçların tamamına ihtiyaç duymadıkları anlatıldı. Kısaca, tüm doğal dil sistemler çözmeye çalıştıkları sorunun bazı yönlerini kolaylaştırmaya çalışırlar. Problem giriş yada çıkış tarafında basitleştirilebilir. Giriş tarafı için, soruyu doğrudan bir veritabanı sistemine yönlendirmek; çıkış tarafı için, belli bir konu hakkında birçok paragraftan oluşan bir gazete metninden üç adet bilgi parçası almak veya tek bir konuşmacının sözlerinden bir görüntüleme sistemi için gereken altı komuttan birini çekip çıkarmak örnek verilebilir. Bu problem basitleştirimleri yukarıda gösterilen bileşenlerden bir yada daha fazlasının basitleştirilmesi veya elenmesi olarak sonuçlanır [8].

Doğal dil işleme sistemleri geliştirilmesindeki ilerleme, muhtemelen eğitime ve değerlendirmeye bağlıdır; ama her birinin kendisine has giriş/çıkış davranışı olan bileşenlerinin çokluğu ve onları uzlaştırmamanın güçlüğü ilerlemeyi zorlaştırır.

Doğal dil işleme problemine diğer bir yaklaşımsa, art arda sıralı kutucuklar yerine, her biri kendisine has bilgi desteği kullanan ve girişin tam olarak anlaşılmasına katkıda bulunan, birbirinden bağımsız süreçler dizisi şeklindedir. Bu mimari Şekil 2.3'teki gibidir [8]. Bu yaklaşımın üstünlüklerinden birisi, yeni ve çok önemli bir bileşenin eklenebilmesine müsaade etmesidir.

Günümüzde doğal dil işlemenin geldiği noktada, milenyumun son altı yılında bu alanda birçok değişim gerçekleşti. İlk olarak, olasılıksal ve “data-driven” modeller doğal dil işleme üzerinde bütünsel bir ölçüt oldu. Ayrışım, sözcük çeşitlerini etiketlendirme, referans çözme ve söylev işleme algoritmaları olasılık içermeye

başladılar ve konuşma tanıma ve bilgi çıkarımından ödünç aldıkları değerlendirme yöntemlerini kullandılar. İkinci olarak, bilgisayarın bellek ve hızındaki artış konuşma ve dil işleme alt alanlarında faaliyet gösteren birçok ticari kuruluştan rağbet gördü (özellikle konuşma tanıma ve imla ve dilbilgisi denetimi alanlarında çalışan kuruluşlar). Konuşma ve dil işleme algoritmaları AAC (Augmentative and Alternative Communication) sahasında kullanılmaya başlandı. Son olarak, Web'in yükselişi dil-tabanlı bilgi çıkarımına şiddetle ihtiyaç olduğunu vurgulamaktadır [8].



Şekil 2.3. Genel bir doğal dil işleme sistemindeki katılımcı sürecin görünüşü

Doğal dil işleme sistemlerinin değerlendirilmesi:

Doğal dil sistemlerinin değerlendirilmesinde güncel olarak kullanılan bir yöntem, bir doğal dil bileşeni içeren sistemin çıkışını gözlemleyerek, onun istenilen çıkış olup olmadığını saptamaktır (bu yöntem ARPA'nın (Amerika) yakın zamanda gerçekleştirdiği konuşma ve dil işleme çalışmalarında başarılı sonuçlar vermiştir). Ama, üretilebilen çıkışların karmaşıklığı ve çeşitliliği, çıkış tabanlı bir değerlendirmeyi güçleştirir [8].

Bu değerlendirmeleri tanımlamak ve gerçekleştirmek yoğun işçilik gerektirir ve de oldukça zordur, ancak bunlar konuşma ve doğal dil işleme alanında araştırma yapan kimseler için çok değerlidirler. Günümüzde bazı doğal dil sistemleri %6'lık hata oranını başarmışlardır. Yakın bir gelecekte bu sistemlerin gerçek uygulamalarda kullanıldıklarına şahit olacağız [8].

Doğal dil işlemenin ne olduğunu anlayabilmek için öncelikle bu alanda kullanılan bazı terimlerin anlamlarının bilinmesi daha uygun olacaktır. Bu terimler şunlardır [7]:

- Fonoloji (Ses Bilim): Yazılı dilde bir ses oluşturan harf toplulukları, ses ya da telaffuz şekli olarak tanımlanabilir.

- Morfoloji (Biçim Bilim): Bir kelimenin biçim bakımından incelenmesini içerir. Kelimenin yapısını ortaya koymaya yöneliktir.

Örneğin “çıkıyorum” kelimesinin,

fonemleri: çı-kı-yor-um;

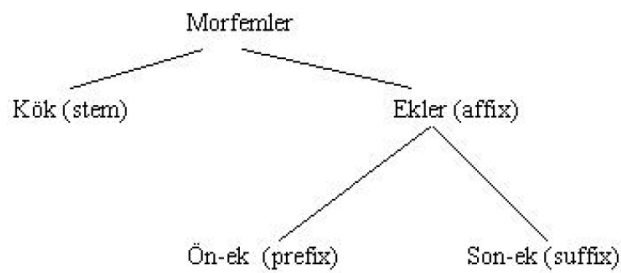
morfemleri: çık-ıyor-um, yani

çık+(Şimdiki Zaman Eki)+(1. Tekil Şahıs) şeklindedir.

- Sözdizim Bilim (Sintaks): Ardarda gelen kelimelerin oluşturduğu yapının belirlenmesidir.
- Anlam Bilim (Semantik): Ardışık olan kelimelerin oluşturduğu anlamın incelenmesidir.
- Söylem (Discourse): Arka arkaya yazılan ve birbiri ile konu ve anlam bütünlüğü olan cümleler içeren metinlere, söylem denilmektedir. Ayrıca metinsel analiz de denilmektedir.
- Makine Çevirisi: Kaynak ve hedef diller arasındaki çeviridir.

## 2.1. Biçim Bilimsel (Morfolojik) Analiz

Kelimelerin, “morfem” denilen (bkz. Şekil 2.1) en küçük anlamlı birimlere ayrıştırılmasına biçim bilimsel analiz denilmektedir [7].



Şekil 2.4. Morfemlerin Yapısı <sup>[7]</sup>

Türkçe sondan eklemeli bir dil olduğundan biçim bilimsel analiz yapılırken eklerin kelimelere getirdiği yeni bir anlam olup olmadığına dikkat edilmelidir. Kimi ekler kelimenin anlamını değiştirmezken, kimi eklerse yeni kelimelerin türemesine neden olmaktadır. Buna göre morfoloji “eklemeli” ve “türetmeli” olmak üzere iki şekilde incelenmelidir. Şekil 2.5 ve Şekil 2.6’te çeşitli örnekler verilmektedir.

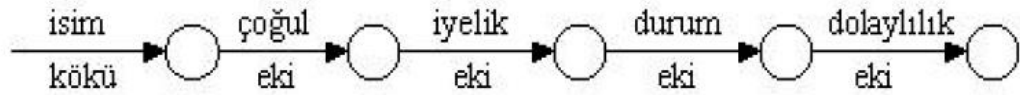
Türkçe bir kelimenin biçim bilimsel analizini yapabilmek için eklerin morfolotik sıralanışının bilinmesi gerekir. Şekil 2.7 ve 2.8’te Türkçe’deki isim ve fiil eklerinin morfolotik sıralanışı verilmektedir [7].

<u>İsme Gelen Ekler</u>	
çoğul ekleri:	masa/masalar
iyelik ekleri:	masam/masan/masası/ masamız/masanız/masaları
İsmin -den hali:	masadan
İsmin -i hali:	masayı
İsmin -e hali:	masaya
<u>Fiile Gelen Ekler:</u>	
Zaman ekleri:	gel/geldi/geliyor/gelmiş/gelecek
Hikaye ekleri:	geliyordum/gelmişti/gelecekti
Zamir ekleri:	geldim/geldin/geldi/ geldik/geldiniz/geldiler

Şekil 2.5. Türkçe’de Eklemeli Morfoloji <sup>[7]</sup>

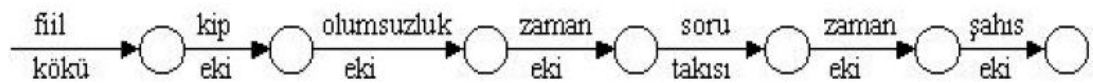
-cı:	kapı/kapıcı
-laş:	uygar/uygarlaş
-mek:	gel/gelmek
-cik:	mini/minicik
-li:	Gebze/Gebzeli

Şekil 2.6. Türkçe’de Türetmeli Morfoloji <sup>[7]</sup>



Şekil 2.7. Türkçe’de İsim Eklerinin Morfolotik Sırası <sup>[7]</sup>

Örnek: oda-lar-ımız-da-ki



Şekil 2.8. Türkçe’de Fiil Eklerinin Morfolotik Sırası <sup>[7]</sup>

Örnek: git-ebil-me-miş-mi-(y) di-ler

Türkçe’de yer alan ortografik bir takım kuralların yine biçimsel analiz sırasında incelenmesi gerekmektedir.

Bu kurallar örneklerle şu şekilde özetlenebilir:

Ünsüz Yumuşaması	→	kazak-kazak(ı)-kaza(ğ)ı,
Ünsüz Benzeşmesi	→	kitaplık-kitaplık(dan)-kitaplık(t)an,
Ekleme	→	kral-k(ı)rala,
Çıkarma (ünlü düşmesi)	→	kayıt(a)-kayda,
Tekrarlama	→	hak-hak(k)a.

## 2.2. Kelime Türünün Belirlenmesi (Part Of Speech- POS Operation)

Bir kelimenin türü ya da diğer bir deyişle ait olduğu sınıf:

- |                                  |                     |
|----------------------------------|---------------------|
| ▲ İsim (Noun),                   | Fiil (Verb),        |
| ▲ Zamir (Pronoun),               | Sıfat (Adjective),  |
| ▲ Zarf (Adverb),                 | Edat (Preposition), |
| ▲ Bağlaç (Conjunction) olabilir. |                     |

Bazı kelimeler bu sınıflardan sadece birine aitken, bazıları bir kaçına birden ait olabilir. Örneğin “yüz” kelimesi, insan yüzü (isim), 100 rakamı (sıfat), yüzmek eylemi (fiil) şeklinde kullanılabilir.

Kelimenin türünün belirlenmesi işlemi, biçimsel analizden sonra, ancak sözdizim analizinden önce yapılmalıdır. Bu işlem, programın çalışması sırasında önceden kelimelerin türleriyle birlikte saklandığı bir veritabanı yapısındaki sözlük vasıtasıyla, bu sözlüğün birden fazla POS’a sahip kelimelerde içinden çıkamadığı durumlarda ise kullanıcıya sorularak da yapılabilir; ya da bir kelimenin cümle içindeki konumu incelenerek o kelime için doğru POS tespit edilerek gerçekleştirilebilir. Bu durumda

bir kelimenin solunda ve sağında yer alan diğerkelimelere bakmak suretiyle geçersiz yorumlar elenerek tek bir POS tespit edilir.

### **2.3. Sözdizim (Sintaks) Analizi**

Sözdizim analizinin amacı bir giriş cümlesini alarak bu cümleyi temsil eden ayrıştırma ağacı denilen hiyerarşik bir yapıyı oluşturmaktır. Ayrıştırma ağacı (anlam ağacı) cümledeki anlamlı birimlere karşılık gelir [7]. Sözdizim analizi için üç bileşene ihtiyaç vardır:

- Sözlük (Leksikon),
- Dilbilgisi (Gramer),
- Ayrıştırma (Parsing) Algoritması'dır.

#### Sözlük (Leksikon):

Sözlük, kök kelimeleri ve her kök kelimenin POS (part of speech) bilgilerini içerir [7]. Proje kapsamında bu amaçla, Microsoft Access ile üretilmiş, veri tabanı biçimde bir sözlük kullanılmaktadır.

#### Dilbilgisi (Gramer):

Bir dildeki cümle türleri ve cümledeki öğelerin diziliş sırası kurallarını içeren formel tanımlamalardır [7].

#### Ayrıştırma (Parsing) Algoritması:

Algoritmanın görevi giriş cümlesini alarak dilbilgisi kurallarına uygun biçimde ayrıştırma ağacını oluşturmaktır. Bir cümlenin bazen birden çok ayrıştırılmış ağacı olabilmektedir. Aşağıdaki stratejilerden biri seçilerek algoritma uygulanır [7].

- ❖ Yukarıdan-aşağıya (top-down, goal-driven) parsing,
- ❖ Aşağıdan-yukarıya (bottom-up, data-driven) parsing,
- ❖ Melez parsing.



## 2.4. Anlamsal (Semantik) Analiz

Anlamsal analiz aşamasının amacı, dilbilgisine göre doğru; ancak anlam olarak geçersiz ayrıştırma ağaçlarının elenmesidir. Bir anlamsal analiz sistemi bir metni analiz ettikten sonra o metinle ilgili soruları cevaplayabilmektedir.

Anlamsal bilginin temsil edilme (representation) yöntemleri şunlardır [7]:

- Önerme Mantığı (First Order Predicate Calculus),
- Kavram Çizgesi (Conceptual Graph /Case Frame),
- Anlamsal Ağ (Onthology),

### 2.4.1. Önerme Mantığı

Bu yöntemde tüm anlamsal bilgiler mantıksal önermeler şeklinde ifade edilir. Matematiksel gösterimlerle ifade edilmesi kolaydır. Ayrıca Prolog dilinde mantıksal önermeler fact'lar olarak kolayca tanımlanabilmektedir [7].

Projede bu yöntem kullanılmıştır. Örneğin;

Ali bir öğrencidir. → ogrenci(ali).

Mustafa Ali'nin babasıdır. → baba(mustafa,ali).

Hava yarın yağışlı olmazsa, Ali dağa çıkar. → hava(yagisli,yarin)=>cikar(ali,dag)

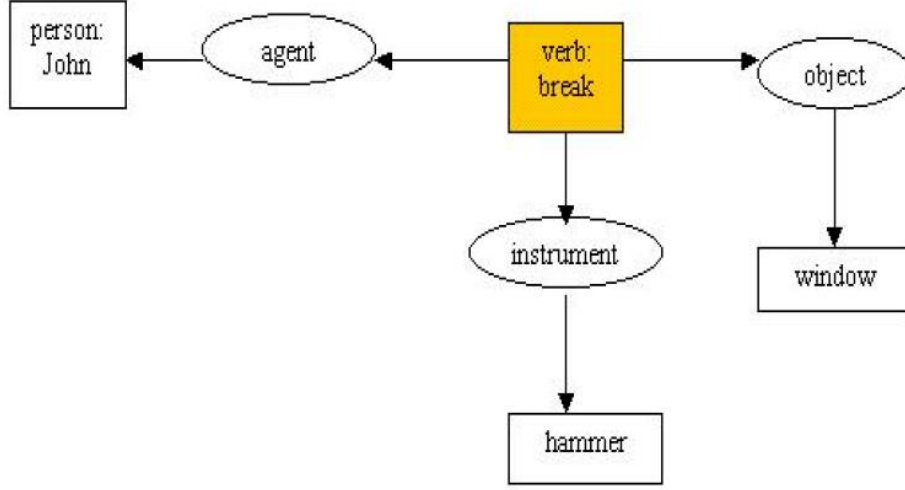
### 2.4.2. Kavram Çizgesi

Öncelikle her fiil için bir kavram çizgesi bir sözlüksel-anlamsal sözlük içinde saklanır. Bir cümle girildiğinde kelimeler kavram çizgesindeki değişken slotlara yerleştirilir, böylece o cümlenin anlamsal temsili elde edilir [7]. Örnek:

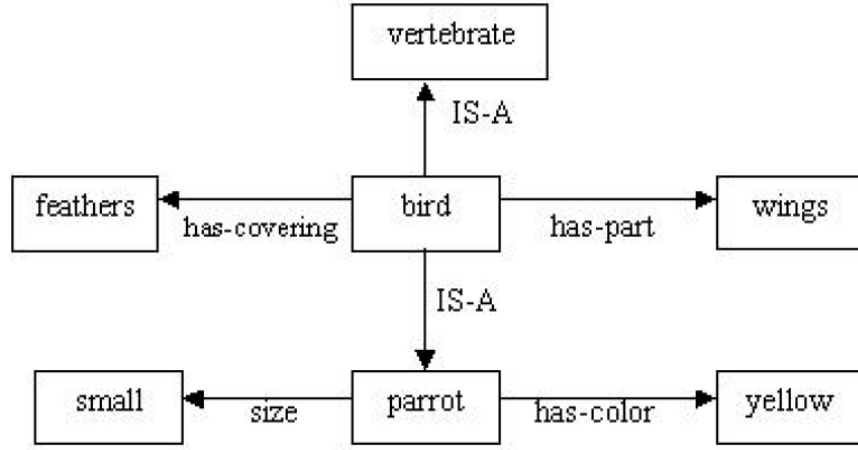
“John broke the window with a hammer” (John bir çekiçe camı kırdı). Bu cümlenin anlamsal gösterimi Şekil 2.9'daki gibidir.

### 2.4.3. Anlamsal Ağ

Anlamsal ağlar genellikle gerçek dünya bilgilerinin temsili amacıyla kullanılırlar. Varlıklar arasında tanımlanan iki önemli bağlantı türü olan “IS-A” ve “HAS-A” kullanılarak bir anlamsal ağ elde edilir [7]. Şekil 2.10'da papağanlarla ilgili anlamsal bir ağ gösterilmektedir.



Şekil 2.9. Anlamsal Gösterim Örneği (Kavram Çizgesi Yöntemiyle) <sup>[7]</sup>



Şekil 2.10. Papağanlara Dair Anlamsal Bir Ağ Örneği <sup>[7]</sup>

Şekil 2.10'deki anlamsal ağın Prolog diline dönüştürülmüş şekli:

isa(bird, vertebrate).	→	<i>Kuş bir omurgalıdır.</i>
hascovering(bird, feathers).	→	<i>Kuş tüylerle kaplıdır.</i>
haspart(bird, wings).	→	<i>Kuşun kanadı olur.</i>
isa(parrot, bird).	→	<i>Papağan bir kuştur.</i>
hascolor(parrot, yellow).	→	<i>Papağan sarı renklidir.</i>
size(parrot, small).	→	<i>Papağanın boyutu ufaktır.</i>

## 2.5. Söylem (Discourse) Analizi

Söylem analizinin amacı, bir metin verildiğinde o metinle ilgili karmaşık soruları cevaplayabilmek, metin içinde doğrudan belirtilmeyen ancak dolaylı olarak anlatılan konuları çıkarsamaktır [7]. Aşağıda James Allen'in tanımladığı söylem fonksiyonlarından bazıları verilmiştir:

- ▲ Konu: Cümlelerin genel olarak ne ile ilgili olduğudur. Genellikle bir cümlelerin ilk ögesi konuyu verir.
- ▲ Açıklama: Cümlelerin yeni bilgi içeren kısmıdır.
- ▲ Odak: Cümlede vurgu yapılan en önemli ögedir. Türkçe'de fiilden önce gelen öge odak kabul edilir.
- ▲ Arka plan: Cümlelerin içinde kullanıldığı bağlamdır.

Örnekler:

O kütüphaneye gitmek ve tüm gün kitap okumak istedi.

*Konu*

*Açıklama*

Kitabını veren Carol idi.

*Odak*

### 2.5.1. Söylem Segmentasyonu (Bölümlenmesi)

Segmentasyon, bir metin içindeki cümle ve ifadelerin aynı konuyu işleyen segmentlere ayrılması işlemidir [7]. Metin anlama ve özetleme uygulamalarında kullanılır. Bunun için bazı ipucu öbekleri ve filtrelerden yararlanılır.

#### İpucu öbekleri

- a. Yeni bir segment başlatan ipucu öbekleri → (şimdi, bu arada, sonra)
- b. Bir segmenti bitiren öbekler → (tamam, güzel, hepsi bu)
- c. Eski bir segmenti devam ettiren öbekler → (herneyse, neyse, böyle, öyle)

#### 2.5.1.1. Stack Kullanarak Söylem Segmentasyonu Yapmak

Algoritma olarak, stack'te en üstteki segment, genişletilmekte olan segment olarak belirlenir. Yeni segmentler stack'e PUSH edilir (eklenir). Biten segmentler stack'ten POP edilir (çıkarılır). Böylece bir önceki segment kaldığı yerden devam ettirilir [7].

Aşağıdaki diyalogta “E” isimli uzman, “A” isimli yardımcıya tamirat konusunda yardım etmektedir [7].

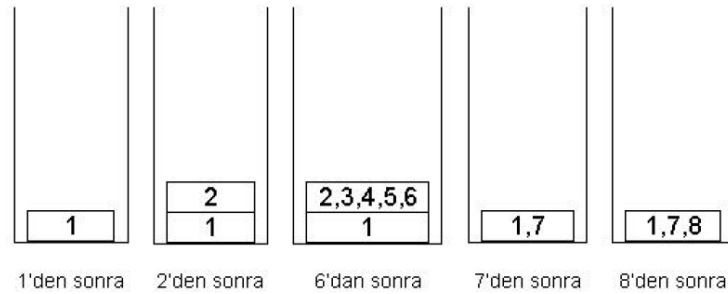
1 E: *Şimdi ipin ucunu motorun tepesine bağla.*

2 *Bu arada bugün benzin aldın mı?*  
3 A: *Evet. Bugün yeni çim biçme makinasını almaya gittiğimde aldım.*  
4 *Ama benzin kutusunu almayı unuttum. O nedenle yeni bir tane aldım.*  
5 E: *Çok tuttu mu?*  
6 A: *Hayır. Ayrıca diğerini de traktörde kullanabiliriz.*

7 E: *Tamam, ne kadar kaldı?*

8 *Bağladın mı şuna?*

Burada 2 -6 arası bir alt diyalogtur. 8'deki “şuna” kelimesinin referans ettiği nesne, 1'deki “motor” kelimesidir. 2'deki “bu arada” ve 7'deki “tamam”, söylem içindeki konu veya odak değişimlerine işaret etmektedir. Şekil 2.11'de stack'in durumu aşama aşama gösterilmiştir.



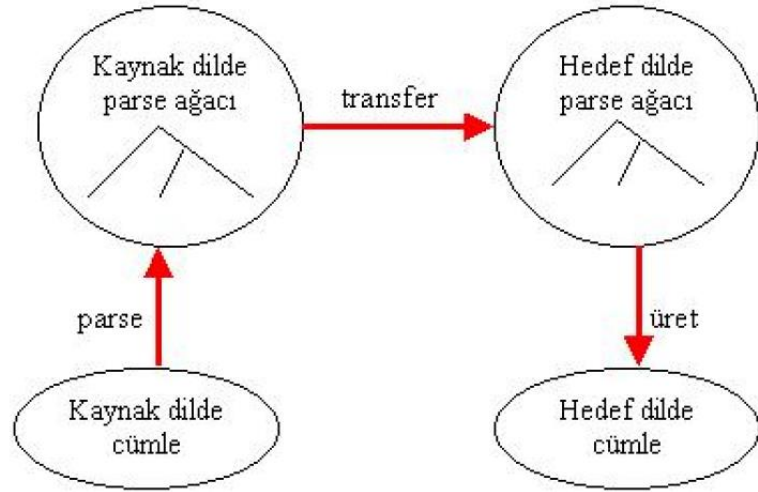
Şekil 2.11. Stack Kullanarak Söylem Segmantasyonu <sup>[7]</sup>

## 2.6. Makine Çevirisi

Bir dilden başka dile otomatik çeviride genellikle iki farklı yaklaşım kullanılmaktadır. Transfer-tabanlı makine çevirisi ve Interlingua-tabanlı makine çevirisidir [7].

### 2.6.1. Transfer-tabanlı Makine Çevirisi

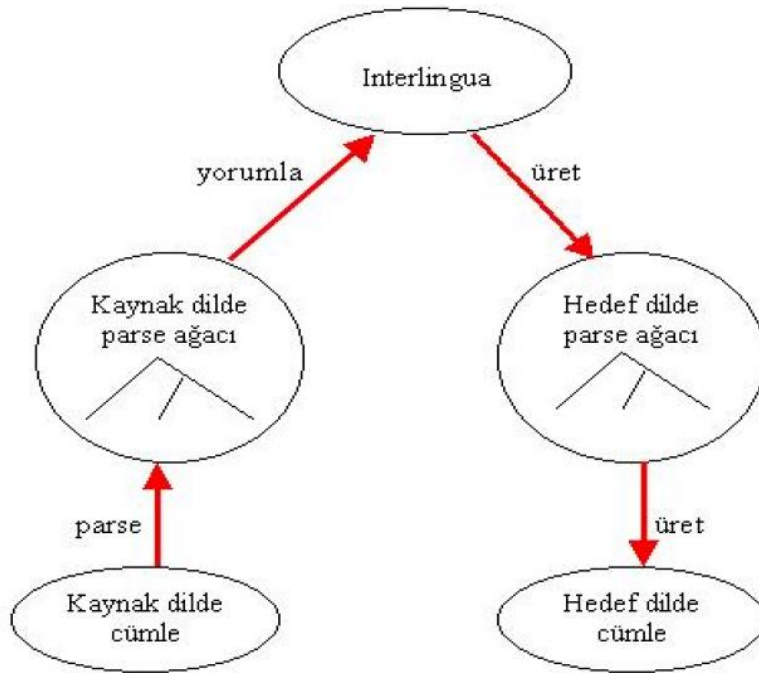
Bu yöntemin işleyişi şekil 2.12'deki gibidir.



Şekil 2.12. Transfer-tabanlı Makine Çevirisi (Projede bu yöntem kullanılmıştır.) <sup>[7]</sup>

### 2.6.2. Interlingua-tabanlı Makine Çevirisi

Bu yöntemin işleyişi şekil 2.13’deki gibidir.



Şekil 2.13. Interlingua-tabanlı Makine Çevirisi <sup>[7]</sup>

### 2.7. Doğal Dil İşlemede Geline Nöka

Veritabanı kullanan soru yanıtlama sistemlerinde soruyu anlama hatası oranı %5 ile %10 arasında değişmektedir. Bir doğal dil sistemini bu hale getirmek için harcanan çaba, hala arzu edilenden fazladır. Bu durum, doğal dil işleme uygulamaları geliştirilmesinin önündeki en büyük engeldir [7].

Taşınabilirlik, bir doğal dil işleme sisteminin yeni bir platformda kullanılmaya elverişliliği olarak tanımlanır. Yeni bir platformda, daha çok otomatik yöntem ve daha az insan emeği kullanmak suretiyle, ılımlı bir başarımla (hata oranı %10-15) yakalayan bir sistem taşınabilir kabul edilir. Taşınabilirliğin amacı, orta derecede bir emekle, iyi bir başarımla elde edilmesidir [8].

Taşınabilirlik sorunu büyük bir ihtimalle birkaç alanda birden yürütülecek bir çalışmayla çözülecektir. Notlar ve ek bilgiler eklenmiş bir sisteme dayalı otomatik öğrenme ise büyük bir gelecek vaat etmektedir. Doğal dil işleme sistemlerinin kullanıcılardan edindikleri yeni bilgilere göre kendilerini yenileyebilme yeteneğine de ihtiyaçları vardır [8].

Taşınabilirliğin yanındaki diğer sorunlar ise gösterilerden gerçek uygulamalara ne zaman geçileceği; sağlamlığın artırılması (umulmayan garip bir girişle sistem nasıl ilgilenecek); geribesleme (yanlış yorumlama durumu oluşursa sistem kullanıcıya nasıl bir yardım önerecek) ve yeni bir doğal dil işleme sistemi için kabul edilebilir başarımla ne olacağıdır [8].

Asıl mükafat makinelerin başarımlarının insaninkinin düzeyinde yada ona yakın olduğu zaman alınacaktır. Doğal dil işleme sistemlerinin önlerinde kat edecekleri uzun bir yol vardır; ama sınırlı alanlarda arzulanı başarmak mümkündür. Sonuçta, sistem tasarımcıları ve geliştiricilerinin (özellikle etkileşimli sistemler) sistemlerine doğal dil işlemeyi dahil etmelerini mümkün kılan ürünler ortaya çıkacaktır. Kullanıcılar kendi sistemlerinden konuşulan yada yazılan komutları ve sorguları anlamalarını, metin gövdelerini sınıflandırmalarını ve bu gövdelerden değişik bilgiler çıkarmalarını bekleyeceklerdir [8].

## **2.8. Türkçe’de Doğal Dil İşleme Alanına Giren Genel Konu Başlıkları**

Türkçe’de doğal dil işleme üzerine yapılan genel çalışmalar şu başlıklar altında toplanabilir [7]:

- ▲ Sentetik Türkçe Sözcük Kökleri Üretimi,
- ▲ Türkçe Dökümanlar İçin Yazarlara Bilgisayar-tabanlı Atıfta Bulunma,

- ♣ Basit Metinlere Yönelik Soru Cevaplama Uygulamaları,
- ♣ Basit Metinlere Yönelik Soru Sorma Uygulamaları ,  
(*Bu Araştırmanın Konusudur.*)
- ♣ Türkçe Konuşma Sentezleyicisi,
- ♣ İstatistiksel Araç Plaka Doğrulama-Düzeltilme Sistemi,
- ♣ Türkçe İçin Okuma Fonksiyonlu Otomatik Metin Oluşturma Sistemi,
- ♣ Metinden Sese Dönüştürme Uygulamaları, (Örnek, “SpeakTRK”)
- ♣ Sesten Metine Dönüştürme Uygulamaları, (Örnek, “IBM ViaVoice”)
- ♣ Diller Arası Kelime ve Cümle Çevirileri (Örnek, “Çevirmen”),
- ♣ Rapor Üretimi,
- ♣ Metin Özetleme,
- ♣ Metin Kategorileme,
- ♣ Kelime-işlem Programları,
- ♣ Bilgi Çıkarma Uygulamalarıdır. (Örnek, IBM WebSphere).

## **2.9. Türkiye’de Doğal Dil İşleme Üzerine Yapılan ve Yapılmakta Olan Projeler**

### **2.9.1. Tamamlanmış Projeler**

Tamamlanmış bazı projeler ve konuları şunlardır [7]:

- TURKISH NATURAL LANGUAGE PROCESSING INITIATIVE/NATO
- Development of Finite State Light Parser for Turkish/TUBITAK
- Development of a Turkish Treebank Corpus/TUBITAK
- Large Vocabulary Continuous Speech Recognition/TUBITAK
- Developing Language Eng.Resources for Low-density Languages/NATO
- Learning Translation Templates from Bilingual Translation Examples Using Machine Learning Techniques/TUBITAK
- Speech Processing Workstation/Gendarmerie Division

- Voice Dialing for Mobile Systems/ASELSAN
- Language pairing on functional structure: LFG-based MT for English-Turkish/ AppTek/Lernout & Hauspie.
- METU Turkish Corpus Project.
- Structure of Turkish Discourse.
- A Grammar Architecture for Computational Analysis of Turkish/TUBITAK.

### **2.9.2. Henüz Tamamlanmış Projeler**

Henüz tamamlanmamış olan bazı projeler ve konuları şunlardır [7]:

- Dependency Parsing with an Extended Finite State Approach
- Large vocabulary discrete speech recognition for Turkish
- Statistical Language Modelling for Turkish
- Information Extraction from Turkish text
- Turkish syntax
- Punctuational devices in NLP, computational semantics and discourse
- Morpho-syntactic generation of surface structure
- Categorical lexicon, morphosyntax-lexicon interface, word order, directionality
- Pro-drop & the lexicon
- Analysis of Turkish discourse, narrative analysis within a cognitive perspective, language acquisition
- Grammar acquisition from Corpus, MT
- Structure of discourse in Turkish
- Parser-Morphemic Lexicon Computational Interface

### **2.10. Doğal Dil İşleme ile ilgili Bazı Önemli Akademik Yayınlar ve Konferanslar**

Uluslararası dergiler [7]:

- Journal of Computational Linguistics



- Journal of Literary and Linguistic Computing

Uluslararası Konferanslar [7]:

- ACL: Annual Meeting of the Association for Computational Linguistics
- EACL: Annual Meeting of the European Chapter of the ACL
- COLING: International Conference on Computational Linguistics
- ANLP: Conference on Applied Natural Language Processing
- EMNLP: Conference on Empirical Methods in NLP

Ulusal Konferanslar [7]:

- TAINN: International Turkish Symposium on Artificial Intelligence and Neural Networks
- SIU: Sinyal İşleme ve İletişim Uygulamaları Kurultayı

### 3. METİN İŞLEME VE ANLAMA TEMELLİ BAZI SİSTEMLER

Tezin konusu doğal dil işlemenin bir alt kolu olan metin işleme ve kullanıcı tarafından girilen metinle ilgili kullanıcıya çeşitli sorular yöneltmektir. Akademik çevreler bu tür sistemlere kısaca şu isimleri vermektedir:

Metin Analizi	→	Text Analysing
Metin Anlama	→	Text Understanding
Metin veya Döküman İşleme	→	Text or Document Processing
Cevap veya Soru Çıkarma	→	Answer or Question Extraction
Soru Yanıtlama veya Sorma Sistemleri	→	Question Answering or Asking Systems

Bu alanda yapılan çalışmalar genellikle yabancı diller üzerinedir. Türkçe üzerine bu tür çalışmalar, daha yeni yeni yapılmaya başlanmıştır. Türkçe’de kelimelerin kök ve eklerine ayrıştırılması, cümlelerin öğelerinin bulunması gibi konularda çalışmalar yapılmakta olup, tezin konusuyla aynı olan bir diğer Türkçe çalışmaya rastlanmamıştır.

#### 3.1. Soru Yanıtlama ve Sorma Sistemlerinin Tarihsel Gelişimi

Bu konuda yazılan en eski makale, 1965 yılında Simmons tarafından yazılan “Answering English Questions by Computer” adlı makaledir [11]. Bilinen en eski soru yanıtlama (SY) sistemleri “Baseball” ve “Lunar” (1973) ‘dır. Baseball sistemi Amerikan liginde bir sezon boyunca oynanan beyzbol maçlarıyla ilgili soru yanıtlamaktaydı. Lunar sistemi ise, ay jeolistlerinin Apollo görevi sonucunda elde edilen, ayın toprak ve kayalarından oluşan bileşimin, kimyasal analiz bilgilerine kolayca erişmesini, bunları karşılaştırmasını ve bu konuyla ilgili çeşitli soruları yanıtlayan bir sistemdi. Her iki proje de ilk başta çocuk oyuncakları şeklinde görülmekteydi. Ancak Lunar’ın 1971 yılında Lunar Science Konferans’ında %78 doğruluk ile soruları yanıtlaması sonucunda bu görüşler tamamen değişti [11].

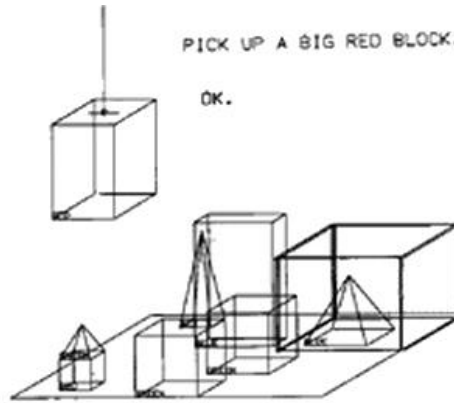
Bunun öncesinde 1966 yılında herkesin bildiği ELIZA diyalog sistemi geliştirildi. Her ne kadar ilk başta büyük sükse yaratsa da ELIZA'nın sadece belli kelimeleri eşleştirip, belli sözcüklerin yerlerini değiştiren, kullanıcının sorularını gerçekten anlamayan bir sistem olduğu anlaşıldı [11].

ELIZA'dan sonra 1972 yılında SHRDLU ve 1977 yılında GUS sistemleri geliştirildi. GUS, bir seyahat danışmanı olarak tasarlandı. Uçak seferleriyle ilgili sınırlı bir veriye erişme hakkına sahipti. SHRDLU ise ELIZA ve GUS'dan daha başarılı bir diyalog sistemiydi ve içinde kutuların ve bir robot elin yer aldığı sanal bir micro-dünya için tasarlanmıştı. SHRDLU'nun nasıl çalıştığını gösteren "demo" aşağıdaki gibidir [11]:

*Pick up a big red block.*  
OK.  
*Grasp the pyramid.*  
I DON'T UNDERSTAND WHICH PYRAMID YOU MEAN.  
*Find a block which is taller than the one you are holding and put it into the box.*  
BY "IT", I ASSUME YOU MEAN THE BLOCK WHICH IS TALLER THAN THE ONE I AM HOLDING.  
OK.  
*What does the box contain?*  
THE BLUE PYRAMID AND THE BLUE BLOCK.  
*What is the pyramid supported by?*  
THE BOX.  
*How many blocks are in the box?*  
FOUR OF THEM.  
*Can the table pick up blocks?*  
NO.  
*Can a pyramid be supported by a block?*  
YES.  
*Can a pyramid support a pyramid?*  
I DON'T KNOW.  
*Stack up two pyramids.*  
I CAN'T.  
*Will you please stack up both of the red blocks and either a green cube or a pyramid?*  
OK.  
*Which cube is sitting on the table?*  
THE LARGE GREEN ONE WHICH SUPPORTS THE RED PYRAMID.  
*Is there a large block behind the pyramid?*  
YES, THREE OF THEM: A LARGE RED ONE, A LARGE GREEN CUBE, AND THE BLUE ONE.  
*Put a small one onto the green cube which supports a pyramid.*  
OK.  
*Put the littlest pyramid on top of it.*  
OK.  
*Does the shortest thing the tallest pyramid's support supports support anything green?*  
YES, THE GREEN PYRAMID.  
*What color is it?*  
BY "IT", I ASSUME YOU MEAN THE SHORTEST THING THE TALLEST PYRAMID'S SUPPORT SUPPORTS.  
RED  
*How many things are on top of the green cubes?*  
I'M NOT SURE WHAT YOU MEAN BY "ON TOP OF" IN THE PHRASE "ON TOP OF GREEN CUBES".  
DO YOU MEAN:  
1 - DIRECTLY ON THE SURFACE  
2 - ANYWHERE ON TOP OF?  
2  
THREE OF THEM.

Thank you.  
YOU'RE WELCOME!

SHARDLU'nun bu diyalogu sonunda çalışma ekranı Şekil 3.1'deki gibidir [11].



Şekil 3.1. SHARDLU'nun Çalışma Esnasındaki Orjinal Ekran Görüntüsü <sup>[11]</sup>

Asıl konu olan metin anlama ya da soru sorma ve yanıtlama sistemleri özellikle çocukların okuduğunu ne kadar anladığının saptanması ve aynı zamanda bilgisayarın girilen metni anlama düzeyinin belirlenmesi açısından çok önemlidir. Bu alanda gerçek anlamda yapılan çalışmalar 1977'li yıllara (QUALM sistemi [11]) rastlamaktadır. Diğer dillerde metin işleme ve anlama üzerine yapılan bazı çalışmalar şu şekildedir:

### 3.2. LILOG (Linguistics and Logic) Projesi

LILOG projesi 1985 yılında "IBM Germany" tarafından doğal dil işlemede anlamsal bilginin Almanca'da işlenmesi amacıyla gerçekleştirilmiştir. Yaklaşık 60 kişinin yıllar süren çalışmaları sonucunda oluşturulmuştur. İlk amaç, yeni bir ürün geliştirmek değil, sadece doğal dil işleme ve bilgi-tabanlı sistemler için yeni teknolojiler geliştirmek amacıyla araştırmalar yapmaktır. Ancak hedef, doğal dil işlemenin en zorlu alanı olan tüm bir metnin anlaşılmasına yöneldi [9].

Metin anlamada karşılaşılan problemlerin büyük bir çoğunluğu, daha basit doğal dil işleme görevlerinin tam doğru şekilde gerçekleştirilememesinden kaynaklanıyordu. Metin anlamının zor oluşunun nedenlerinden biri, dil bilimi ile mantığın birlikte kullanılmasıydı. Daha teknik bir terimle, hesaplamalı dil bilim ile yapay zekanın harmanlanmasıydı. Bu nedenle projeye LILOG adı verildi. Burda ana amaç bilgisayarın Almanca girilen bir metni anlamasını sağlamaktır. Peki ama bir bilgisayarın metni anlayıp anlayamadığı nasıl anlaşılacaktı? Çözüm çok klasikti. Bir

öğretmen, öğrencisinin metni anlayıp anlamadığını nasıl kontrol ediyorsa o şekilde, yani sisteme metinle ilgili sorular yöneltilerek. Bu durumda bilgisayar hem girilen metni, hem de soruları anlayıp, bunlara doğru cevap vermek zorunda kalacaktı [9].

### 3.2.1. LILOG Sistemi Ne Yapar?

LILOG sisteminin şu anki versiyonu olan LEU/2 Almanya'nın Düsseldorf şehrinin turist rehberindeki aşağıdaki paragrafı okur:

*Im Palais Nesselrode ist das Hetjensmuseum, das 1909 eröffnet wurde, untergebracht. Es befindet sich an der Ecke Schulstraße und Hafenstraße. Die Keramiksammlung umfaßt zehntausend Objekte. Der Eintritt der Ausstellung, die von 10 Uhr bis 17 Uhr geöffnet ist, beträgt 2 DM [9].*

[1909'da açılan Hetjens Müzesi Palais Nesselrode'da bulunmaktadır. Burası Schulstrasse ve Hafenstrasse'nin köşesindedir. Seramik koleksiyonu on bin parçadan oluşmaktadır. Sergiye giriş, açık olduğu 10 a.m. ile 5 p.m. arasındadır.]

Bu metin işlendiğinde gerekli bilgiler sistemin bilgi tabanına çeşitli eklerle kaydedilir. Metindeki her bir kelime, sistemin sözlüğünden incelenir ve biçimsel, dilbilgisel ve anlamsal bilgi, ilk cümle için dil bilgisi analizi yapacak olan ayrıştırıcıya gönderilir. İlk cümle için çeşitli anlamsal işlemler yapılır ve cümlelerin içeriği bilgi sistemindekiyle ilişkilendirilir. Son olarak, cümlelerin mantıksal bilgiye dönüştürülmüş şekli sistemin hafızasına yerleştirilir. Diğer cümleler için de aynı işlemler tekrarlanır [9]. Bir metni anlamamanın önemi şu şekilde görülebilir:

Sisteme sorulan soru:

*Wann hat das Hetjensmuseum geöffnet?*

*[Hetjens Müzesi ne zaman açıktır?]*

Sistemin yanıtı:

*Von 10 Uhr bis 17 Uhr.*

*[10 a.m.ile 5 p.m. arası]*

Sistem bu sonuca nasıl vardı? Metin müzenin açık olduğu saatleri açıkça belirtmemiştir. Sistem serginin seramik koleksiyonu sergisi olduğunu, bu seramik koleksiyonunun müzenin bir parçası olduğunu, bu nedenle serginin açık olduğu zamanlarda, müzenin de açık olması gerektiğini anlamıştır [9].

Ayrıca bu sistemde kullanıcının daha önce sorduğu sorular da izlenebilmelidir. Aksi takdirde kullanıcının üstteki diyalogdan sonra, “2 p.m. de açık olur mu?” sorusunda kastettiğinin müze olduğunu anlayamaz.

LILOG metin anlama sistemi başka şeyler de yapabilmektedir. Örneğin bir harita yardımıyla, Düsseldorf’ta bir A noktasından B noktasına nasıl gidileceğini kelimelerle yazılı olarak anlatabilmektedir [9].

### **3.2.2. LILOG Sisteminin Yapamadıkları**

Eğer sistem tarafından tüm girilen metinler anlaşılabilir duruma gelirse, sistem “*İşsiz biri kamp için yaptığı harcamaları gelir vergisinden düşebilir mi*” sorusuna yanıt verebilir. Belki de sistem, hangi durumlarda gelir vergisi indirimleri olacağı konusunda kullanıcıya detaylı bilgiler sunabilir.

Metin anlamada karşılaşılan iki rahatsız edici durum vardır. Bunlardan biri, sadece dil ile ilgili bilgilerin yeterli olmaması, bunun yanında metnin ilgili olduğu konuyla ilgili bilgilerin de sistemde bulunması zorunluluğudur. Örneğin rehberlik konusunda bilgili olan bir sisteme, kanuni bir metin verildiğinde anlamsal analiz süreçleri yetersiz kalacaktır. Aksi takdirde Almanca bilen herkes kanuni metinleri eksiksiz anlayabilirlerdi.

Sistemi bu alanlarda bilgi sahibi yapmak o alandaki bilgi mühendislerinin işidir. Bu da sistemin iş yükünü ve maliyetini arttıracaktır.

Sistemde bir diğer eksik nokta sağduyu ile kazanılan bilgilerdir. Şu an sıradan bir insanın ne derece büyük bir sağduyu bilgisine sahip olduğu ve bu bilginin sisteme nasıl verileceği bilinmemektedir. Ayrıca sağduyu ile kazanılan bilgilerin sürekliliği de değişmektedir. Sistemin bu sorunları, bu gün her türlü doğal dil işleme sisteminde var olan sorunlardır.

### **3.3. Bir Cevap Çıkarım Sistemi: ExtrAns**

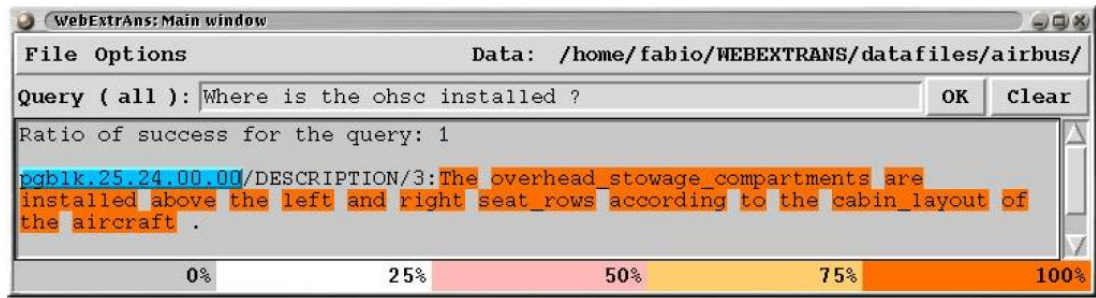
ExtrAns, bir cevap çıkarım sistemidir. Cevap çıkarım sistemleri bir metinle ilgili kullanıcı sorularını yanıtlayan sistemlerdir. Bu sistemlerle ters şekilde çalışan soru çıkarım sistemleri de bulunmaktadır. Ancak, bu sistemlerin tasarımı biraz daha karmaşıktır. Her iki sistemde de “metnin boyutu” ve “taşınabilirlik” hayati önem taşımaktadır. Boyut büyüdükçe sistemlerin başarı oranı düşmektedir.

ExtrAns, Unix sayfalarını ayrıştırabilmekte ve bu cümlelerin mantıksal formunu oluşturabilmektedir. Özellikle Unix'in "online" el kitabı için tasarlanmıştır. Örneğin "Unix man pages". Ayrıca kullanıcı soruları da mantıksal forma dönüştürülebilmektedir. Sistemin hassasiyetini arttırmak için tüm sistemi değil, sadece "dilbilgisi" ve "anlam çıkarım" bileşenlerini güçlendirmek yetmektedir [10].

ExtrAns'ın en temel özelliği metinsel bilgiyi mantıksal bir forma dönüştürebilmesidir. Bunu yaparken, kelimeler ve cümleler arasındaki ilişkileri saptamak üzere metin içindeki fiilleri, isimleri, sıfatları, bağlaçları, edatları ve zamirleri kullanmaktadır [10].

Bu sistem Unix'in el kitabı için hazırlandığından küçük ölçekli bir metinle çalışılmıştır. Kullanıcının sorularının basit bir İngilizce'yle sorulması istenmektedir.

Şekil 3.2'de ExtrAns sisteminin çalışmasından bir örnek verilmektedir.



Şekil 3.2. ExtrAns Sisteminin Sorgulama Ekranı

### 3.3.1. Geri Çekimli Arama Stratejisi

Bu strateji nedeniyle Unix'in el kitabı için bir "eş anlamlılar sözlüğü" oluşturulmuştur. Buna göre Unix el kitabı kavramı için oluşturulan bu sözlükte tüm ilişkiler "eş anlam" ve "eş ses" üzerine kuruludur [10].

Arama algoritması şöyle çalışmaktadır: Kullanıcı sorusundaki tüm sözcüklerin (eş anlamlılar sözlüğünde olanların) eş anlamlıları, başlangıçtaki kullanıcı sorgulamasına eklenir. Eğer sorgulama yeterli sonuçla geri dönmezse, tüm sözcüklerin eş seslileri de sorguya eklenerek arama (sorgulama) işlemi tekrarlanır. Eğer bu da yeterli sonuçla geri dönmezse, son çare olarak sistem "keywords" yani "anahtar kelimeler" moduna geçer. Bu modda kullanıcı sorgusundaki tüm isim, sıfat, fiil ve zamirler seçilir ve bunların aynen geçtiği cümleler metin içinde aranır. Sorgulamanın uzunluğuna göre elde edilen sonuçlar da değişmektedir [10].

### 3.3.2. ExtrAns Sisteminde Geliştirilmesi Gereken Noktalar

ExtrAns henüz yeterli düzeyde değildir ve çalışmalar yapılmaktadır. Sistemin gerçekten yararlı olması için Unix'in 30 sayfalık metinlerinden daha büyük metinleri analiz edebilmesi sağlanmalıdır. Sistemin eksilteli cümlelerle, çeşitli deyimlerle, ve söyleyişlerle çalışabilmesi, tam olmayan cümleleri de analiz edebilmesi sağlanmalıdır. Sistemde anlamsal analiz güçlendirilmeli, bazı cümlelerde ortaya çıkan uzun anlamsal formlar sadeleştirilmeli ve karışıklık giderilmelidir. Ayrıca cümleleri bağlayan bağlaçlara karşı önlem alınmalıdır [10].

### 3.4. LOLITA (Large-scale, Object-based, Linguistic Interactor, Translator, and Analyser) Sistemi

LOLITA, genel amaçlı bir DDİ sistemi olarak tasarlanmıştır ve 1986'dan bu yana Durham Üniversitesi'nde sürekli geliştirilmektedir. Sistem farklı platformlarda DDİ uygulamalarını sağlamak amacıyla tasarlanmıştır. Bunu, üzerinde farklı uygulamaların oluşturulabileceği çekirdek bir platform oluşturarak yapmaya çalışmaktadır. Bu çekirdek platform iki temel modüle sahiptir: metinleri mantıksal forma dönüştüren, metnin anlamını veren "analiz" modülü ve mantıksal formdaki ifadeleri bir metne dönüştüren "üretim" modülüdür. Çağdaş DDİ uygulamalarının tersine LOLITA, farklı alanlarda yeniden biçimlendirilebilecek bir framework yapısında tasarlanmamıştır. Sistem sadece belli alanlarda çekirdek bir yapı olarak kullanılabilir [12].

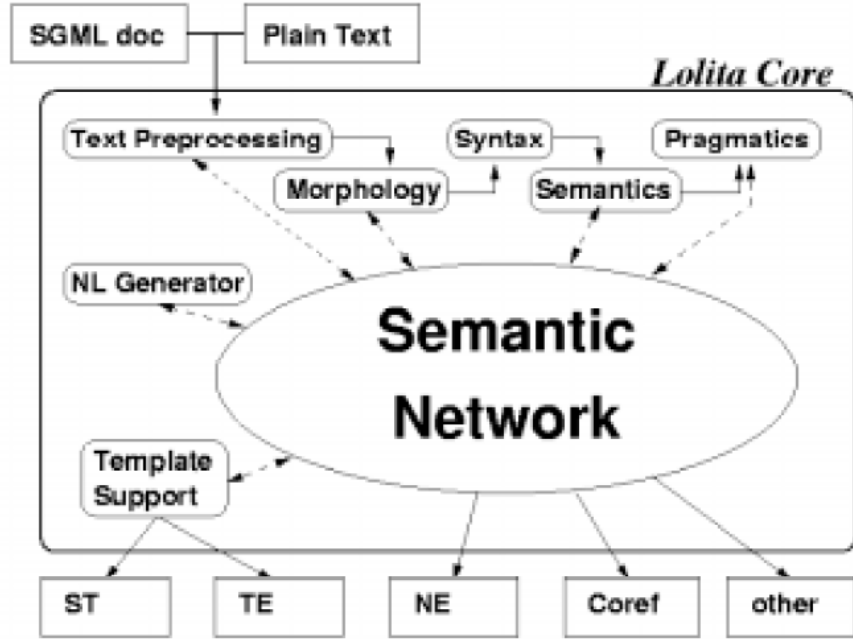
Durham Üniversitesi'nde bu çekirdek platform kullanılarak çeşitli prototip uygulamalar gerçekleştirilmiştir. Bunlardan bazıları:

- Bilgi ve Özet Çıkarma,
- Doğal Dilde Sorgulama ve Cevaplama: LOLITA'ya bilgi verilip daha sonra bu bilgiyle ilgili sorular sorulması,
- Bir Diyalog Modelinin Oluşturulması,
- Çince Dilbilgisi Kurallarına Yönelik Öğrencileri Sınayan Bir Uygulama Geliştirme,
- İngilizce Problem Teşhisi Yapan Uygulama Geliştirme.



### 3.4.1. LOLITA Sistemi'nin Mimarisi

Şekil 3.3 'te LOLITA Sisteminin çekirdek yapısının bazı uygulamalarla olan etkilişimi görülmektedir.



Şekil 3.3. LOLITA Sisteminin Blok Diyagramı <sup>[12]</sup>

LOLITA bunu destekleyen uygulamalar dizisiyle uyum şekilde çalışacak bir çekirdek olarak yaratılmıştır. Şekil 3.3'deki uygulamalar MUC'da (*Message Understanding Conference*) belirtilen uygulamalardır. Çekirdeğin en önemli parçası “büyük bilgi sistemi” dir. Buna “Semantic-Netwok-SemNet” adı verilmektedir. Analizin her aşamasında SemNet yoğun bir şekilde kullanılmaktadır. Ayrıca analiz sonucunda elde edilenler sürekli bu bilgi tabanına eklenerek dinamik bir yapı oluşturulmuştur [12].

Çekirdeğe bağlı uygulamalar analiz sonuçlarını SemNet'ten okuyabilirler. Bunu gerçekleştirmek için SemNet'e doğrudan erişim hakkına sahiptirler ve SemNet'i sorgulayarak istedikleri sonucu elde edebilirler. Bu yardımcı uygulamalar yazımda yardımcı olan, yazım hızını arttıran uygulamalar ya da SemNet'teki bilgileri İngilizce'ye dönüştüren doğal dil üreticiler olabilir [12]. LOLITA'nın mimari yapısını daha detaylı anlayabilmek için M.H. Smith'in “Natural Language Generation in LOLITA System” adlı 1995 yılında Durham Üniversitesi'nde yazdığı doktora tezi incelenebilir [12].

### Ön –işlem

LOLITA sisteminde metin işleme yapılmadan önce metinler, (HTML ya da normal metinler) SGML ağacı yapısına dönüştürülmek üzere bir ön-işleme tabi tutulurlar. Bunu SGML ayrıştırıcı yapmaktadır. Eğer SGML ağacına eklenmesi gereken ek bilgiler varsa bunlar da ağaca eklenir. İlk önce paragraflar, sonra cümleler sonra da kelimeler halledilir [12].

### Morfoloji

SGML ağacına morfoloji uygulanır. SGML ağacında yapraklar farklı sözcük token'larını, düğümler ise dökümanın yapısını temsil eder. Düğümdaki yapılar karışıklık meydana getirmemesi için açılır. Örneğin "I'll"; "I will" haline getirilir. Temel morfoloji fonksiyonu ağacın tüm yapraklarına uygulanır. Sözlüğe bakılarak kelimelerin kökleri bulunur ve bunlar lexical ve semantic düğümlere bağlanır. Bir sözcüğün birden fazla anlamı varsa tüm anlamlar o yaprağın alt düğümlerinde tutulur [12].

### Parsing (Ayrırma)

Parsing işlemi Tomita algoritmasına göre yapılmaktadır [13].

#### **3.4.2. LOLITA Sistemi'nin Oluşumu**

LOLITA, çok yaygın olmayan bir programlama dili olan Haskell ile yazılmıştır. Çok kritik olan parsing ve SemNet algoritmaları ise C dilinde yazılmıştır. Haskell esnek bir dil olmasından dolayı seçilmiştir ve yapı itibarıyla LISP'e çok benzemektedir.

LOLITA'nın başarısı, çekirdek bir sistem olduğundan çok önemlidir. Eğer başarısı kötü olursa, onun üstüne kurulmuş tüm sistemlerin başarısı da kötü olacaktır. Şu andaki LOLITA sisteminin üç temel eksiği bulunmaktadır [12]:

- Parsing işlemi yeterince iyi değildir. En son gelişmelerden sonra dahi %20'lik bir hata söz konusudur.
- İsmi olan varlıkların algılanması düşük düzeydedir ki bu sorun SemNet bilgi tabanına yeni şirket adları, model kodları v.s. ekleyerek çözülebilir.
- Program kodu bazı önemsiz hatalardan (özellikle de SGML ağacı oluşturulurken) henüz tamamiyle arınamamıştır.

#### 4. METİN İŞLEME ve ANLAMA TEMELLİ ARAŞTIRMALARIN UYGULAMA ALANLARI ve GELECEĞİ

Metin işleme ve anlama temelli araştırmaların günümüzde uygulanabileceği alanlar ve gelecekte ne gibi amaçlarla kullanılabileceğine dair bir takım fikirler öne sürülebilir.

##### 4.1. Uygulama Alanları

Metin işleme, anlama, sorgulama ve soru soran ve yanıtlayan sistemlerin temel geliştirilme nedeni, kullanıcı dostu sistemler geliştirmek ya da var olan sistemleri kullanıcıya daha yakın kılarak, kullanıcının adeta bir insanla konuşur gibi karşısındaki bilgisayarla iletişimini sağlamaktır. Bu nedenle geliştirilen sistemlerin bazı kullanım alanları şu şekilde özetlenebilir:

- Web üzerinden kullanıcıların sorularını yanıtlayan sistemler. Bu tür sistemler arama motorları sistemlerinden daha üst sistemlerdir ve kullanıcının aradığını daha kolay bulabilmesini sağlarlar. Yeni nesil arama motorları bu sistemlerden oluşacaktır. Örneğin, “*Telefon ne zaman icat edildi?*” sorusuna yanıt verebilen bu sistemlere karşılık, şu andaki arama motorlarında bu sorunun yanıtı için anahtar kelimelerin ve sonuç olarak kullanıcıya dönen bir çok sitenin çok iyi analiz edilmesi ve cevabın bulunması gerekmektedir.

Web üzerinden soru yanıtlayan sistemlere örnekler: *Ask Jeeves*, *SHAPAQA*, *MULDER* soru yanıtlama sistemleridir [11].

- Örneklerde de verdiğimiz gibi bir şeyin tamiri sırasında kullanıcının aklına takılan soruları bilgisayara sormasıyla tamir işlemini hatasız tamamlamasını sağlayan karşılıklı diyalog sistemleri. Diyalog sisteminin bir diğer örneği de çok bilinen ELIZA sistemidir. Piskoloji alanında bilgi tabanına sahip metin anlama tabanlı diyalog sistemleri ile kişilerin bilgisayarla konuşup, rahatlaması sağlanabilir. Bu tür sistemlere uzman sistemlerin de eklenmesiyle, bu konuşma sonucunda hasta ya da kullanıcı için çeşitli

teşhisler ya da kendisine yardımcı olacak çeşitli yorumlar bilgisayar tarafından üretilebilir.

- Herhangi bir alan için özel olarak hazırlanmış, girilen metni anlayıp, metinle ilgili kullanıcı sorularını yanıtlayabilen sistemler. Örneğin, belli bir bölge hakkında gerekli metinsel bilgiye sahip bir sistemin, gelen turistlerin sorularını yanıtlaması, onlara rehberlik etmesi ya da kanuni alanda bir bilgi tabanına sahip sistemin belli kanuni işlemler konusunda, bir avukat gibi kullanıcıların sorularına yanıt vermesi.
- Özellikle metin anlama tabanlı sistemlerde, metnin özetinin çıkarılması. Örnek: GETARUNS sistemi [14]. Bu sistem ayrıca soru yanıtlama görevi de görmektedir.
- Metin anlama tabanlı soru yanıtlama ve soru sorma sistemlerinin yaygın olarak kullanılabileceği bir alan da eğitimidir. Ancak şu andaki gelişme bu sistemlerin ilköğretim seviyesinde kullanılabilmesine olanak tanımaktadır. Öğretmen öğrencisinin anlamasını istediği metni bilgisayara girer. Daha sonra bilgisayar bu metni daha önce bahsedilen aşamalardan geçirerek mantıksal forma dönüştürür. Bundan sonra sistem iki farklı şekilde tasarlanabilir.

Eğer sistem soru yanıtlama sistemi olarak tasarlanmışsa, öğrencinin metinle ilgili sorularına yanıt vererek zaman kaybedilmeden istenen veriye ulaşması sağlanarak öğrencinin öğrenme sürecine katkıda bulunur.

Sistem soru sorma sistemi olarak tasarlanmışsa, öğrenciye metinle ilgili ilköğretim düzeyini aşmayacak şekilde sorular yöneltir ve öğrencinin cevabını yine mantıksal forma dönüştürerek, kendi cevabıyla karşılaştırır. Eğer yanıt doğru değilse, doğru yanıtı öğrenciye yine normal bir cümle olarak sunar. Bu tezde tasarlanmak istenen sistem de aynen bu şekilde çalışmaktadır. Temel amaç, ilköğretim düzeyindeki öğrencileri “okuduğunu anlama” konusunda sınava tabi tutmaktır.

#### 4.2. Geleceğe Yönelik Kullanım Alanları

Henüz bu konuda gelinen nokta yeterli değildir. Şu anki sistemler genelde tek bir döküman için analizler gerçekleştirmektedir ve kullanılan dökümanlar hep ufak boyuttadır. Gelecekte bir çok dökümanı işleyebilen, çoklu bilgi tabanı sistemlerine ve büyük dökümanları işleme yeteneğine sahip ya da internet üzerinden bu bilgi tabanlarına bağlantılar sağlayıp, mevcut dökümanları tarayabilen sistemler geliştirilmesi planlanmaktadır. Bu da yeni nesil arama motorları geliştirilmesine olanak sağlayacaktır. Aranılan bilgilere daha kolay erişim imkanı doğacaktır. Böyle bir çalışma örneği olarak “AnswerBus” sitesi örnek verilebilir [15]. Bu site tam anlamıyla kullanıcıya direkt yanıtlar vermemekle birlikte, yanıtı içeren ve yine bir arama motoru gibi çeşitli site bağlantıları (link ler) sunmaktadır. Farklı dillerde soru sorma imkanı da sunmaktadır.

Bir diğer hedef ise, farklı dökümanlardan elde edilen benzer; fakat farklı yanıtların kullanıcıya bir özet olarak sunulması, diğer bir deyişle, metin işleme ve özetleme tekniklerinin birleştirilmesidir. Bu durumda kullanıcının kafası karışmamış olacaktır.

Bir diğer çalışma, gelecekte bu sistemlerin çoklu diller için de kullanılabilir olmasını sağlamaktır. Örneğin eğer bu çalışma bir sonuca ulaşırsa, İngilizce bir metin ile ilgili sorular, Türkçe sorulup, bu soru bilgisayar tarafından İngilizce’ye çevrilip, metinsel sorgulama yapıp, tekrar Türkçe yanıt kullanıcıya sunulabilecektir.

Anlamsal analiz çalışmaları daha da geliştirilerek, bilgisayarın metinde bire bir geçmeyen bilgiler hakkında yorum yapıp, kullanıcıya cevap vermesi sağlanmaya çalışılmaktadır. Örneğin,

*VERİ: “Ali ve Mehmet aynı yıl doğdular. Mehmet 1980 yılında doğdu.”*

*SORU: “Ali ne zaman doğdu?”*

*Yanıt: “1980 yılında doğdu.”*

Geleceğe yönelik yapılan araştırma konularından biri de metin içindeki atıflardır. Dijital bir atıf standartı oluşturulup, bu standartta atıf yapılarak yazılmış metinlerin soru yanıtlama amacıyla yapılan analizinde, atıf yapılan dökümanların da internet vasıtasıyla incelenmesi ve kullanıcıya daha detaylı bilgi vermesi mümkün olan sistemler geliştirilmek istenmektedir.

Tüm bu gelecekteki çalışmalar özellikle akademik çevrelerde çok büyük ilgi görecektir. Çünkü akademisyenler, araştırmacılar ve bilim adamları zamanlarının büyük bir kısmını istedikleri bilgiye ulaşmaya çalışmakla harcamaktadır. Soru sorma ve soru yanıtlama gibi sistemler sayesinde istenen bilgiye ulaşmak için harcanan zaman oldukça kısılacak, bilgiye erişim kolaylaşacak ve kişiler, ilgi alanlarına giren konularla daha çok ilgilenebilecek zamanı bulabileceklerdir.

## 5. UYGULAMAYA YÖNELİK TEMEL BİLGİLER

Bu temel bilgiler genel olarak Türkçe dilbilgisi temel kuralları ve uygulamanın hazırlandığı platform hakkında yardımcı bilgiler sunan teknik bilgiler olarak gruplanabilir.

### 5.1. Türkçe Dilbilgisi Temel Kuralları

Türkçe, Ural-Alтай dil grubuna giren, sözcük yapısı ve üretimi açısından bitişik bir dildir. Bu tip dillerde sözcükler bir kök sözcüğe, sanki üzüm taneleri gibi eklenen birimlerden oluşurlar. Bu birimler eklendikleri kök veya gövdenin anlamını, sözcük türünü veya sözdizimsel işlevini değiştirebilirler [6].

#### 5.1.1. Türkçe’de Kök ve Ek İlişkisi

Sözcük çekimleri; zaman ve basit köklerden yeni sözcüklerin türetilmesi, sözcük köklerinin sonuna ek ya da eklerin bitleştirilmesiyle gerçekleşir. Kök-ek bitişmesinde, yabancı sözcüklerin de dilimize girmesiyle bazı istisnai durumlar olmakla birlikte, sıra ve biçimi belirleyen kesin kurallar bulunmaktadır. Türkçe’deki bir kelime genel olarak aşağıdaki yapıdan oluşmaktadır [6].

*Sözcük kökü + yapım ek(ler)i + çekim ek(ler)i*

Yapım ekleri, sözcük köklerine bitişerek sözcüğün hem anlamını hem de türünü değiştirirler. Çekim ekleri ise sözcüğün anlamını değiştirmemekle birlikte, türünü değiştirebilmektedir. Fakat hiçbir çekim eki isim soylu sözcüğe bitişerek fiil soylu sözcük ya da fiil soylu sözcüğe bitişerek isim soylu sözcük türetemez; yalnızca bir addan zamir türetmek gibi, isim soylu sözcüklerin alt grupları arasında tür geçişi yapılabilir [6].

#### 5.1.2. Çekim Eklerinin Kök ya da Gövdeye Bitişi

Türkçe’de yer alan tüm sözcükler, isim soylu ya da fiil soyludur. İsim soylu sözcüklere örnek olarak; isim çekimleri, bileşik isimler v.s.’yi sayabiliriz. Fiil soylu

sözcükler de kökü veya gövdesi fiil olan ve fiil çekim eklerinin herhangi birisini alan sözcüklerdir.

#### 5.1.2.1. Fiil Çekim Ekleri

Fiil çekim ekleri; kurallı bileşik fiil ekleri, zaman ekleri ve şahıs ekleri olmak üzere üç kısma ayrılabilir. Zaman ekleri ise kurallı zaman ekleri ve kip ekleri olmak üzere iki ana gruba ayrılmaktadır. Aynı zamanda, bileşik fiillerin bulunması ile ilgili kurallar da çalışmaya dahil edilmiştir.

#### 5.1.2.2. İsim Çekim Ekleri

Türkçe dilbilgisinde, genel olarak bir isim çekimi şu şekilde olmaktadır [6]:

*İsim + (ÇoğulEki | -) +İyelikEkleri + HalEkleri +(EkFiilEkleri | KiEki | -) + ŞahısEkleri*

Yukarıdaki kuralda; “ | ” (ya da) ayırıcı, içine alınan ekin seçmeli olduğunu göstermektedir. “ÇoğulEki” kelimenin birden fazla olduğunu belirten ekleri, “İyelikEkleri” aitlik belirten ekleri, “HalEkleri” ismin hallerini belirten ekleri, “EkFiilEkleri” -idi, -imiş, -dir gibi ekleri, “KiEki” dolaylı anlatımda kullanılan -ki ekini ve “ŞahısEkleri” ise şahıs (kişi) eklerini göstermektedir.

### 5.2. Prolog

Prolog mantıksal programlamanın bir temsilcisidir ve prosedürel programlama dilinden daha çok bir veritabanına benzeyen tek dildir. Mantıksal teorem temelinden geliştirilmiş ve başlangıçta doğal dil işlemedeki araştırmalarda kullanılmıştır.

Popüleritesi esas olarak uzman sistemler, doğal dil ve akıllı veritabanları gibi uygulamalar için düzenlenen AI (Yapay Zeka) komitesinde yayılmasına rağmen, geleneksel uygulama tipleri içinde kullanışlıdır. Birçok dile göre daha hızlı geliştirmeyi ve prototip oluşturmayı sağlar [2].

Prolog’da programlama önemli şekilde geleneksel prosedürel programlamadan farklıdır ve programlamaya farklı bir tarz getirir. Bu dilde mantıksal ilişkiler açıklanır ve Prolog, belirtilen ifadelerin doğru olup olmadığına ve eğer doğruysa ifadeyi hangi değişken bağlayıcılarının doğru yaptığına karar verir. Prolog’da program kodunun yazımı ve bakımı daha kolaydır.



Programcılar kontrol yapıları ve kontrol mekanizması hakkında endişelenmek zorunda değildirler. Bu kontrol işlemleri otomatik olarak Prolog tarafından yapılır. Bununla birlikte, programcı programın istenilen şekilde çalışması için Prolog'un "built-in" kontrol yapılarını yönetebilir [2].

Prolog'un güçlü arama ve model eşleme özelliği vardır. Arama mekanizması için geriye iz sürme metodunu izler, yani programın işlemesi hem ileri hem de geriye gidebilir. Diğer dillerde olduğu gibi özyineleme (recursive) mekanizmasını destekler ve özyineleme Prolog uygulamalarında önemli bir yere sahiptir [2].

Prolog, bu sahip olduğu niteliklerden dolayı özellikle doğal dil sistemlerini geliştirmek için çok uygundur.

### **5.2.1. AMZI! Prolog**

Amzi!'nin temel teknolojisi, en başarılı ve önemli mantıklı düşünme araçlarından biri olan Prolog dili ele alınarak, güçlü ve taşınabilir özellikte yaratılmıştır. Logic Server (Mantıksal Sunucu) veritabanları ve istemcilerdeki/sunuculardaki bilgileri muhakeme etme yeteneğine sahiptir. Buna ek olarak Logic Server herhangi ekstra arabirimler yada kapasiteler sağlamak için genişletilebilir [2].

#### **5.2.1.1. Mimarisi**

Amzi!'nin yapısının çekirdek kısmında derlenmiş (statik) ya da yorumlanmış (dinamik) Prolog kodunu yükleyen ve çalıştıran "runtime" bir makina vardır. Uygulama geliştiriciler Etkileşimli Geliştirme Ortamını (IDE=Interactive Development Environment) ya da komut satırı araçlarını kullanarak Prolog programlarını, mantık tabanlarını yazabilir, test edebilir, hatalarını giderebilir (debug), derleyebilir [2].

Derlenmiş program Prolog makinası tarafından yüklenebilen .xpl uzantılı ikili (binary) dosyalarda saklanır. Yorumlanmış kod, metin dosyalarında saklanabilir ve herhangi bir Prolog programı tarafından da okunabilir [2].

#### **5.2.1.2. Logic Server Engine ve LSAPI**

Logic Server basit olarak bir dinamik kütüphane (.dll) gibi paketlenmiş Prolog runtime makinasıdır. Prolog Engine iki C++ nesnesidir : Biri makina için, diğeri de exception handling ( harici durumları kontrol etmek) içindir. Bu nesneler, Logic

Server API (LSAPI) 'nın fonksiyonlarına uygun olan “public” metodlara sahiptirler. LSAPI, windows programcıları için “amzi.dll” olarak kullanılabilir. Link yapmak için kullanılan “amzi.lib” ile birleştirilmiştir [2].

#### **5.2.1.3. Delphi Bileşeni**

Amzi! Logic Server Delphi bileşeni (TLSEngine) Delphi programcısına Logic Server DLL tarafından sağlanan servislere kolay erişim imkanı vermek için tasarlanmıştır. Logic Server bileşeni “amzi.pas” dosyasında kaynak yapısında verilmiştir.

Uygulamada Logic Server kullanmak için, basit olarak Logic Server bileşeni Component Palette'in Additions sayfasından forma yerleştirilir. Tipik olarak bu bileşen ana formun bir parçası haline gelir. Bu işlem Unit'in Uses bölümünde 'Amzi' yi dahil ederek 'LSEng :TLSEngine' gibi Logic Server için bir değişken yaratılarak elle de (manual) yapılabilir. Logic Server'ı çağırmak için basit olarak, API çağrıları uygulamanın metodlarına yerleştirilir [1].

#### **5.2.1.4. Amzi! Prolog'un Desteklediği Ortamlar**

Masaüstü bilgisayarlar için Amzi! Prolog ve Logic Server Windows 95, 98, NT 4.0, 2000 ve XP'de kurulabilir. Desteklediği bütün ortamlarla ilgili kütüphaneleri içerecek şekilde Amzi! sisteminin kurulumu için yaklaşık 6,5 MB'lık disk alanı gerekmektedir.

Logic Server API'sı şunları destekler [2]:

- Microsoft Visual C++, Borland C++, Java, Delphi, Visual Basic, MS-Office : Access, Excel, Word
- Web Server'ları, bir dinamik library 'leri (.dll) çağırabilen herhangi bir windows geliştirme tool'u.

#### **5.3. Dll ( Dynamic Link Library ) Dosyaları**

Dinamik bağlantılı kütüphaneler olarak tanımlanan DLL dosyaları ile, içerisinde program kodu ve veri barındıran kaynakların, gerek Windows uygulamaları, gerekse bir çok program tarafından paylaşımı sağlanmaktadır. Bu paylaşım, program çalışırken yapılabilmekte ve DLL dosyaları sayesinde aynı program kodlarını kullanan uygulamalarda, DLL dosyalarının bu sırada hafızaya yüklenmesiyle, bu

birimin daha hızlı bir biçimde kullanılması sağlanmaktadır. Aynı zamanda hafızadan da tasarruf edilmiş olmaktadır. Örneğin ortalama hesaplayan bir DLL dosyası oluşturulduktan sonra, ortalama hesabının gerektiği her bir programda bu DLL dosyasına direkt ulaşılabilir. Bunun için DLL dosyasının bulunduğu yer, DLL'in kullanılacağı programda belirtilmelidir. ISAPI teknolojisiyle oluşturulan DLL dosyalarının kullanımı biraz daha farklıdır. Bölüm 5.3.1'de daha ayrıntılı bilgi verilmektedir.

### **5.3.1. ISAPI ( Internet Server API ) / NSAPI ( Netscape Server API ) Genel Yapısı**

Dinamik web sayfalarının oluşturulabilmesi için en sık kullanılan iki protokol CGI (Common Gateway Interface) ve web hizmet birimi API'lerdir. Üçüncü bir yöntem olan ASP (Active Server Pages) da son derece popüler hale gelmiştir [1].

Web hizmet biriminin API'larını kullanan iki farklı API vardır. Bunlar, ISAPI (Microsoft tarafından sunulan Internet Server API ) ve NSAPI (Netscape Server API)'dır. Bu API'lar hizmet biriminin kendi adres uzayına yüklediği ve genellikle bir süre için hafızada tuttuğu bir DLL yazılmasına olanak tanır. Hizmet birimi DLL'i yükledikten sonra, her ortamda taşınabilir olan DLL uzantılı bu dosya ile programlar EXE'siz çalıştırılabilir.

Hizmet birimi yeni bir sayfa talebi aldığı anda, bu DLL'i yükler ve sayfa talebini işlemek için yeni bir kanal başlatabilecek olan gerekli kodu çalıştırır. DLL kodu daha sonra sayfayı talep etmiş olan istek birimine, gereken verileri gönderir. Bu iletişim genellikle hafızada gerçekleştiği için, bu tür bir uygulama genellikle daha hızlı çalışır.

DLL'in dezavantajlarından biri, DLL hafızada olduğu takdirde yenilenmiş ya da değiştirilmiş bir sürümünün derlenememesidir. Bunun için ilk önce DLL'in hafızadan silinmesi ya da Web hizmet biriminin ( PWS ya da ISS ) geçici bir süre için kapatılması ve yeniden başlatılması gerekir.

#### **5.3.1.1. Delphi WebBroker ve WebModule Teknolojileri**

Delphi yapısında bulunan VCL, Web üzerinde server tabanlı geliştirmeyi basitleştirmek amacıyla oluşturulmuş, bir sınıf hiyerarşisi olan WebBroker'ı ve WebModule adlı özel bir veri modülü tipini kullanır [1].

WebBroker ile ISAPI ve CGI uygulamaları çok kolaylıkla hazırlanabilir. Bunun için Object Repository'nin ilk sayfasında, "WebServer Application" simgesi seçilir. Bundan sonra karşımıza üç alternatif çıkar : ISAPI, CGI ve WinCGI. Bunlardan ISAPI seçilir. Böylece bir ISAPI uygulamasının temel yapısı Delphi tarafından yaratılır. Bu oluşturulan uygulama WebModule yapısına dayalıdır. En önemli işlemler WebModule içinde gerçekleştirilir [1].

WebModule yapısı içinde, gelebilecek talebin yol adına bağlı olarak bir Actions dizisi kullanılır ve bu dizi, kullanılan WebModule'ün Actions editörü ile oluşturulabilir. Örneğin "project1.dll" için, "ilk" adlı bir action oluşturulmuşsa bu action'a ulaşmak için :

*"http://localhost/scripts/project1.dll/ilk "* şeklinde bir url adresi kullanılabilir.

Bu şekilde farklı uygulamalar hazırlayarak, farklı isteklere yanıt verilebilir. Bir çok action, aynı DLL dosyasında tutulabilir. DLL dosyaları bir kez çağrıldıktan sonra belleğe yerleştikten, DLL dosyası bellekteyken farklı isteklerde bulunulduğunda bu isteklere hızlıca cevap verilebilir [1].

OnAction Event'i ile, belli bir isteğe verilecek yanıt için gerekli olan kod oluşturulur. Bunun için **Response** parametresinin **Content** özelliği kullanılır. Böylece kullanıcıların görmesi gereken sayfaların HTML kodları girilebilir.

### 5.3.1.2. Neden ISAPI Teknolojisi?

Tez çalışmasında ISAPI teknolojisinin kullanılması uygun görülmüştür. ISAPI teknolojisi ile DLL dosyaları, Delphi diline çok benzer bir şekilde oluşturulabilmektedir. Internet üzerinden "cümlelerin analizi ve öğelerine ayrılması" projenin ilk adımlarından birisidir. Amaç, bu uygulamanın Internet üzerinden herkese açık olmasını sağlamaktır. Bu nedenle, Delphi'nin de desteklediği, herkese açık bir Internet ortamı sağlayan ISAPI ve DLL teknolojilerini kullanmak, uygun olur düşüncesiyle, ISAPI teknolojisinin kullanılmasına karar verilmiştir. Ayrıca DLL dosyalarının kendilerini belleğe yerleştirerek hızlı bir şekilde çalışmaları da bu düşüncede etkin bir rol oynamıştır. Delphi'nin baz olarak alınmasının nedeni ise tezi yazanın (Zeki Mocan) en hakim olduğu dil olmasından dolayıdır.

## 6. SORU SORAN SİSTEM UYGULAMASININ GERÇEKLEŞTİRİLMESİ

### 6.1. Temel Adımlar

Uygulamada izlenen adımları kısaca şu şekilde özetlemek mümkündür:

- I. Kullanıcının girdiği metin içindeki kelimeler kök ve eklerine ayrılır.
- II. Eklerden yararlanılarak cümleler, temel ve yan cümlelere ayrılır.
- III. Temel ve yan cümleciklerin öğeleri bulunur.
- IV. Öğelere ayrılmış olan cümleler Prolog'a uygun formata dönüştürülüp, önermeler olarak Prolog'a gönderilir.
- V. Prolog'a gönderilen önermeler sorgulanarak cümlelerde bulunan öğelere göre sorular üretilir ve uygun formata dönüştürülerek kullanıcıya yansıtılır.
- VI. Kullanıcının, soruya verdiği cevap için de ilk 4 adım uygulanır.
- VII. Kullanıcının verdiği cevap ile eldeki Prolog önermeleri, sorulan soruya uygun şekilde sorgulanarak cevabın doğruluğu kontrol edilir. Cevap yanlışsa, doğru yanıt kullanıcıya sunulur.

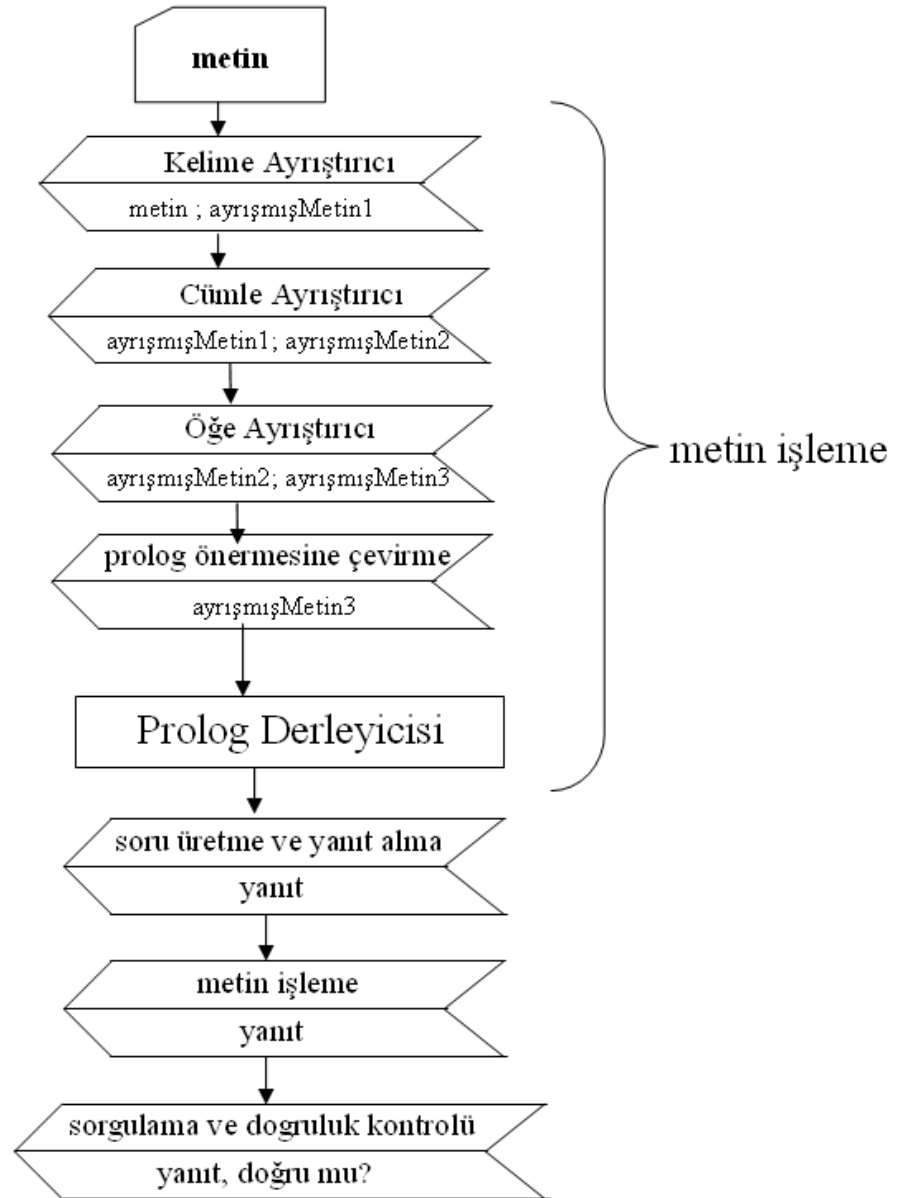
Şekil 6.1'de sistemin genel akış diyagramı verilmektedir.

#### 6.1.1. Kelime Ayırıştırıcı (Kelimeparser)

Bu fonksiyon, girilen metni eklerine ayırır, her kelimenin ilk karakterinden başlayarak (soldan) ve her seferinde bir karakter ekleyerek bu kelimenin içindeki kökleri (yeni türemiş olan köklerle birlikte) bulur. Daha sonra kelimenin aldığı eklere ve cümle içindeki durumuna göre olabilecek ihtimallerden birini seçer. Sonuçta metin içindeki her kelimeyi *kök + çekim ekleri* şeklinde bir yapıya dönüştürür.

#### 6.1.2. Cümle Ayırıştırıcı (Cumleparser)

Kelime ayırıştırıcıdan geçirilmiş metindeki cümleleri, eğer birden fazla eylem (fiil, fiilimsi) içeriyorsa tek eylemli basit cümleler haline dönüştürür ve bu bileşik cümlelerin basit cümleleri arasındaki ilişkileri ve bağlantıları belirler.



Şekil 6.1. Sistem Akış Diyagramı

### 6.1.3. Öge Ayırıştırıcı (Ogeparser)

Kelime ve cümle ayırıştırıcılarından geçmiş olan, tek eylemli basit cümlelerin öğelerini bulur.

### 6.1.4. Prolog Önermesine Çevirme

Öğelerine ayrılmış basit cümleleri, Prolog'a uygun önermeler haline dönüştürür ve bu önermeleri Prolog'a gönderir. Bu sırada Amzi! Prolog'un desteklediği Mantıksal Sunucu Motoru'ndan yararlanır.

### **6.1.5. Soru Üretme**

Prolog önermelerini herhangi bir öğeye göre sorgular ve eğer o öğe, önermeden bir değer döndürürse, bu sorgulamayı doğal dile dönüştürerek kullanıcıya soru olarak iletir.

### **6.1.6. Yanıt İşleme**

Kullanıcının soruya verdiği yanıtı, aynı işlemlerle Prolog önermeleri haline dönüştürür.

### **6.1.7. Doğruluk Kontrolü**

Soru üretme modülünün, soruyu oluştururken kullandığı sorgulamayı, yanıt önermeleri için uygular ve aynı cevabı alıp-almadığını kontrol ederek yanıtın doğruluğuna karar verir.

## **6.2. Kelimelerin Kök ve Eklerine Ayrılması**

Kelimelerin kök ve eklerine ayrılması işlemi sırasında; ilk aşama olarak kelime içindeki bütün olası kökler, bir Türkçe sözlükten bulunmaktadır. Bu şekilde, her kelime için birkaç tane olası kök elde edilmektedir ve her olasılık için de kelimenin geriye kalan kısmının çekim eklerinden meydana gelmesi gerektiği gözönünde bulundurularak, olasılıklar sözdizimsel ve anlamsal olarak incelenerek gerçek çözüm belirlenir. İkinci adımda ise fiilimsilere göre cümleler, temel ve yan cümlelerine ayrılmıştır (Bkz. bölüm 6.3). Daha sonra elde edilen bu temel ve yan cümleciklerin öğeleri bulunmuştur (Bkz. bölüm 6.4).

### **6.2.1. Ek ve Köklerine Ayırma Modülü**

Kelimenin cümle içindeki durumuna bakılarak kök ve eklerine ayrılması; doğal dille yazılmış metinler üzerinde çalışan bir sistemin önemli bir bileşenidir. Türkçe sondan eklemeli bir dil olduğundan dolayı; Türkçe üzerinde doğal dil işleme çalışmalarında sözcüklerden eklerin atılması denildiğinde, kastedilmek istenen hep son eklerin atılması olmuştur.

Gövdeleme algoritmalarında temel olarak iki yöntem izlenmektedir. İlk yöntemde, kelime soldan sağa doğru harf harf eklenerek veri tabanında aranmaktadır. Diğer yöntem ise kelimenin sağdan sola doğru harf harf kırılarak incelenmesidir. Bu

çalışmada, bütün olası kökler bulunacağından, kelimenin harf sayısı kadar inceleme yapılacaktır. Bu nedenle iki yöntem aynı etkinliği göstermektedir. Bu çalışmada ilk yöntem tercih edilmiştir.

Sistemde, ilk olarak kelimenin olası tüm kökleri bulunmaktadır. Örneğin; “ağacı” kelimesi için “ağ”, “ağa” ve “ağaç” kelimeleri kök olarak bulunmaktadır. Kökler bulunduktan sonra iki farklı durum söz konusudur. Eğer kök, ‘isim soylu’ ise kelimenin içerdiği isim çekim ekleri araştırılır. Şayet fiil soylu ise ‘fiil çekim ekleri’ ve fiilimsi ekleri araştırılır. Kökün hangi soydan olduğu sözlükte belirtilmektedir. Bu işlemler sonucunda elimizde birden fazla çözüm bulunmaktadır. Bu çözümlerden bir kısmı, kelimenin tamamını içermediğinden dolayı zaten elenir. Geriye kalan olası çözümler içerisinden doğru çözümü bulurken ise kelimenin cümle içerisindeki durumuna bakılır. Örneğin, “evi” kelimesi için “ev” kök olarak ve “-i” eki ise ya ‘hal eki’ ya da ‘iyelik eki’ olarak bulunur. Eğer cümlede “Ayşe’nin evi” benzeri bir tamlama varsa bunun iyelik eki olduğu anlaşılır. “Şu evi beğendim” cümlesinde ise “-i” ekinin hal eki olduğu anlaşılır.

#### **6.2.1.1. Olası Köklerin Bulunması**

Bu adımda kelime, metindeki durumundan bağımsız olarak tek başına incelenmektedir. Kelimenin harfleri soldan sağa doğru birer birer eklenerek elde edilen kelimenin sözlükte olup-olmadığı araştırılır. Burada harf harf ilerlerken dikkat edilmesi gereken nokta, “sessiz benzeşmesi” ve “sessiz yumuşaması” olaylarının gerçekleşmesi ihtimalidir. Bu nedenle her yeni harf eklendiğinde oluşan bu harf dizisinde benzeşme ya da yumuşama olma ihtimaline bakılır. Örneğin “ağacı” kelimesi için yumuşama olayını düşünmeden bulunabilecek kökler “ağ” ve “ağa” dır. Oysa yumuşama ihtimalini de göz önüne alarak, gerçek bulunması gereken kök olan “ağaç” kökü de bulunur. Diğer taraftan kelime kökünde yardımcı ses ve kaynaştırma harfinin olması mümkün olmadığından bu durum incelenmemiştir.

#### **6.2.1.2. Çekim Eklerinin Bulunması**

Bu adımda, kök bulma modülü sonucu elde edilen kelimenin isim soylu mu yoksa fiil soylu mu olduğuna göre (ki bu sözlükte belirtilmiştir) ya *isim\_cekim\_bul* ya da sırasıyla *kuralli\_bilesik\_fiil* ve *fiil\_cekim\_bul* modüllerine gönderilir.



## İsim\_çekim\_bul Modülü

Bu modül üç alt fonksiyondan oluşmaktadır :

- a) İlk fonksiyonda çekim eki olarak çoğul ekinin olup-olmadığı araştırılır. Eğer çekim eki varsa ve kelime sonu ise kelime tamamlanmıştır. Kelimenin sonuna gelinmemişse ikinci fonksiyona gönderilir. Bu birinci fonksiyonda dikkat edilmesi gereken nokta, “arabaları” örneğinde olduğu gibi “-ler” ekinin çoğul eki değil de”-leri” 3. çoğul iyelik eki olduğu durumdur. Bu duruma çözüm bulmak için “-ler” çoğul eki bulunsa bile, çoğul eksiz kelime direk olarak ikinci fonksiyona gönderilir.
- b) İkinci fonksiyonda ise, iyelik çekim eklerinin olup-olmadığı araştırılır. Eğer varsa ve kelimenin sonuna gelinmişse o zaman cevaba ulaşılmıştır. Buradaki önemli nokta, eğer 3. tekil şahıs iyelik eki bulunmuşsa ve kelimenin sonuna gelinmişse bunun aynı zamanda “-i” hal eki olabileceği durumudur. Programda bu da göz önünde bulundurulmuştur. İyelik eki bulundu ise ama kelimenin sonu değilse üçüncü fonksiyona gönderilir. Aynı zamanda “-m”, “-miz” iyelik eklerinde olduğu gibi bulunan iyelik eki seçiminin doğru olmaması ihtimali sebebiyle, sanki iyelik eki bulunmamış gibi tekrar indis üzerinden ikinci fonksiyona gönderilir. Ayrıca iyelik eki hiç bulunmadıysa da direk üçüncü fonksiyona gönderilir.
- c) Üçüncü fonksiyonda ise ilk önce bu tüm kelime içerisindeki “-ki” eki içeren bir yapı olup-olmadığı araştırılır. Daha sonraki işlem, hal eklerinin (“-e”, “-i”, “-de”, “-den”, “-le”, “-in”) bulunmasıdır. En son adım, bulunan hal eklerine bağlı olarak son eklerin bulunmasıdır. Son eklerin bulunmasında ise aşağıdaki adımlar izlenir :
  - i. Şahıs ekleri araştırılır. Eğer ek bulunamadıysa ii. adıma geçilir. Eğer ek bulunduysa zaten kelimenin sonudur.
  - ii. Ekfiil zamanları (“-dir”, “-se”, “-miş”, “-di”) araştırılır. Ek bulunamadıysa iii. adıma geçilir. Eğer ek bulunduysa, ek fiilden sonra şahıs eki gelme ihtimali olduğundan şahıs ekleri de araştırılır.
  - iii. “-ken” çekim ekinin varlığı araştırılır. Eğer ek bulunamadıysa kelime gövdesinin yanlış seçildiği anlaşılır.

## **Kurallı\_bilesik\_fiil Modülü**

Türkçe’de dört çeşit kurallı bileşik fiil bulunmaktadır. Bunlar; yeterlik fiili, tezlik fiili, sürerlik fiili ve yaklaşma kurallı bileşik fiilleridir. Bu fonksiyonda kelimenin kurallı bileşik fiil eklerinden bir tanesini alıp-almadığına bakılır. Aynı zamanda kelimenin olumsuzluk eki olan “-me”, “-ma” ekini alıp-almadığı da bu modülde incelenmektedir.

## **Fiil\_cekim\_bul Modülü**

Türkçe’de bir fiil gövdesine maksimum iki tane zaman eki ve bir tane şahıs eki yani toplam üç çeşit ek gelebilir. Buna göre; bu zaman ekinin ve çekim ekinin geliş sırası, gövde+zaman+şahıs, gövde+zaman+zaman+şahıs ya da gövde+zaman+şahıs+zaman şeklinde olabilmektedir. Ayrıca fiil gövdesi hiçbir zaman sadece şahıs eki alamaz.

Fiil\_cekim\_bul modülünde izlenen yol, kelimenin sonuna gelinene kadar kelimeye yukarıda belirtilen üç çeşit ek bulunana kadar ya da hiç ek bulunmayana kadar zaman ve şahıs eklerinin aranmasıdır. Ek bulundukça, ek gelmiş kelime kaydedilir. Ayrıca ara adımlarda sessiz benzeşmesi, yumuşaması, kaynaştırma harfi, koruyucu ünsüz ve yardımcı ses olaylarının da kontrolü yapılır.

Buna ek olarak; türemiş bir fiil gövdesinin fiilimsi (isimfiil, sıfatfiil, zarffiil) olma ihtimali bulunduğundan dolayı bu kelimeye fiilimsi ekinin olup-olmadığı da ayrı bir modül halinde incelenir.

## **6.2.2. Analiz Modülü**

Kelimenin kök ve eklerinin ayrıştırılması işlemi sonucunda, çeşitli şekillerde ek ve köklerine ayrılmış kelime dizileri elimizde bulunmaktadır. Bunlardan kimisi zaten kelime boyuna eşitlenmediği için ara işlemler sırasında elenmiştir. İşte analiz kısmında, bunun dışında kalan, kök ve eklerin çeşitli kombinasyonları sonucu birleşirken oluşturdukları ama Türkçe dilbilgisi yapısına uygun olmayan kelime dizileri elenecektir. Eleme işlemi sırasındaki kriter, kelime dizisindeki eklerin birleşme sırasında birbirlerine olan uyumluluklarıdır. İnceleme sırasında işlem yine soldan sağa doğru yapılmaktadır

Eğer elde edilen “string”, isim soylu ise analiz modülü sırasıyla *fiilimsi\_eki\_analizi* ve *isim\_cekim\_ekleri\_analizi* parçalarından; eğer fiil soylu ise *fiil\_cekim\_ekleri\_analizi* kısmından oluşmaktadır.

#### **6.2.2.1. Fiilimsi\_eki\_analizi Modülü**

Bu fonksiyonda, bir fiilimsi ekinin doğru olarak kelimeye bitişmesi için gerçekleşmesi gereken kurallar çıkarılmıştır ve bitişme olayının bu kurallara uyup-uymadığı analiz edilmiştir. Bu kurallar aşağıdaki gibidir:

- a) “-r” sıfat fiil ve zarf fiil eki için, “-r” ekinden önceki kelime gövdesinin son harfi ya kaynaştırma harfi ve koruyucu ünsüz dışındaki herhangi bir sessiz harf ya da yardımcı ünsüz olmalıdır.
- b) “-r” eki dışında sessizle başlayan fiilimsi ekleri için; bu ekten bir önceki ek kaynaştırma harfi, koruyucu ünsüz ya da yardımcı ses olmamalıdır.
- c) Sesli ile başlayan fiilimsi ekleri için; bu ekten önceki kelime gövdesinin son harfi ya kaynaştırma harfi ve koruyucu ünsüz dışındaki herhangi bir sessiz harf ya da “y” kaynaştırma harfi olmalıdır.

#### **6.2.2.2. İsim\_çekim\_ekleri\_analizi Modülü**

Bu fonksiyonda, her isim çekim eki için bitişme kuralı çıkarılmıştır. Kelimenin sonuna kadar, teker teker bulunan bu isim çekim ekleri alınarak, ilgili kurala uyup-uymadığına bakılır ve uymuyorsa elenir. Bu kurallar aşağıdaki gibidir :

- a) Çoğul eki için:

Bu ekten önceki ek, kaynaştırma harfi, yardımcı ses ya da koruyucu ünsüz olmamalıdır.

- b) İyelik ekleri için:

- 3. tekil iyelik eki için, bu ekten önceki kelime gövdesinin son harfi ya kaynaştırma harfi ve koruyucu ünsüz dışındaki herhangi bir sessiz harf ya da “s” kaynaştırma harfi olmalıdır.
- 3. çoğul iyelik eki için, bu ekten önce kaynaştırma harfi, koruyucu ünsüz ya da yardımcı ses gelmemelidir.
- Diğer iyelik ekleri için, bu ekten önceki kelime gövdesinin son harfi sesli olmalıdır.

- c) Hal ekleri için :

- Yönelme (“-e”) ve belirtme (“-i”) hal eki için, bu ekten önceki kelime gövdesinin son harfi ya kaynaştırma harfi ve koruyucu ünsüz dışındaki herhangi bir sessiz harf ya da “y” ve “n” kaynaştırma harflerinden biri olmalıdır.
  - Bulunma hali ve ayrılma hali için, daha önceden iyelik eki almış bir kelime ise ve bu ek, iyelik eki üçüncü şahıs eki ise, o zaman bu hal eklerinden önce “n” koruyucu ünsüzü gelmelidir. Diğer durumda yani iyelik eki almamış durumda ise sadece bu eklerden önce kaynaştırma, yardımcı ses ya da koruyucu ünsüz gelmemelidir.
  - Tamlayan eki (“-in”) için, bu ekten önceki kelime gövdesinin son harfi ya kaynaştırma harfi ve koruyucu ünsüz dışındaki herhangi bir sessiz harf ya da “n” kaynaştırma harfi olmalıdır.
- d) Ki’ li yapıdaki her “-ki” bloğu için sırasıyla aşağıdaki analizler yapılır :
- Varsa yukarıda belirtildiği gibi hal ekleri analizi yapılır.
  - Varsa “-ken” eki analizi yapılır.
  - “-ki” ekinden bir önceki ekin yardımcı ses, koruyucu ünsüz ya da kaynaştırma harfi olmaması gerektiği araştırılır.
  - Varsa çoğul eki analizi yapılır.
- e) Ek fiilin zamanı için :
- Ek fiilin geniş zamanı için, bu ekten bir önce kaynaştırma, yardımcı ses ya da koruyucu ünsüz gelmemelidir.
  - Diğer zamanlar içinse bir önceki ek ya koruyucu ünsüz ya da koruyucu ünsüz dışındaki diğer sessiz harflerden bir tanesi ile biten bir ek olmalıdır.
- f) Ek fiilin şahıs eki için :
- 1. tekil ve çoğul şahıslar için, bir önceki ek ya “y” kaynaştırma harfi ya da bu kaynaştırma harfi dışındaki diğer sessiz harflerden bir tanesi ile biten bir ek olmalıdır.
  - Diğer şahıs ekleri için ise, bir önceki ek yardımcı ses, koruyucu ünsüz ya da kaynaştırma harfi olmamalıdır.

### 6.2.2.3. Fiil çekim ekleri analizi Modülü

Bu fonksiyonda, şahıs eklerinin ve zaman eklerinin bitişme kuralları belirtilmiş ve ekin bu kurallara uyup-uymadığına bakılmıştır. Bir fiil çekim ekinin eklendiği kelimede uyması gereken kurallar aşağıdaki gibidir :

- a) Eğer fiil çekim eki; gelecek zaman eki, istek kipi eki, ikinci şahıs emir kipi veya görülen geçmiş zaman ekinin, öğrenilen geçmiş zaman ekinin ve dilek-şart kipinin bileşik fiilleri ise; bu ekten önceki kelime gövdesinin son harfi ya kaynaştırma harfi ve koruyucu ünsüz dışındaki herhangi bir sessiz harf ya da “y” kaynaştırma harfi olmalıdır.
- b) Gereklilik kipi, üçüncü şahıs emir kipi veya görülen geçmiş zamanın, öğrenilen geçmiş zamanın ve dilek-şart kipinin birinci zamanı ise bu eklerden bir önceki ek kaynaştırma harfi, koruyucu ünsüz ya da yardımcı ses olmamalıdır.
- c) Şimdiki zaman veya geniş zaman eki ise bu ekten önceki kelime gövdesinin son harfi sesli olmalıdır.

### 6.2.3. Kelimenin Cümle İçerisindeki Durumunun İncelenmesi

Bu modülde; analiz modülü sonucu elde edilen ve Türkçe dilbilgisi kurallarına uygun olan; ancak kelimenin cümle içerisindeki durumuna göre birden fazla olabilecek çözümlerden esas çözümün bulunması gerçekleştirilmektedir. Analiz modülünden çıkan birden fazla çözüme örnek olarak, “Evi çok güzel” ile “Evi gördün mü?” cümlelerindeki “evi” kelimesini verebiliriz. İlk cümlede “evi” kelimesindeki “-i” eki üçüncü tekil şahıs iyelik eki, ikinci cümledeki “-i” eki ise hal ekidir. İşte bu nedenle, “evi” kelimesi için, analiz modülü sonucunda “isim-3.tekilİyelik” ile “isim-HalEki” çözümleri elde edilir ve gerçek çözümün bulunması için “evi” kelimesinin cümledeki işlevini incelemek gerekir.

İşte, bulunan birden fazla sonucun indirgenmesi için kelimenin cümle içerisindeki durumuna bakılacaktır. Yalnız bu işlemin doğru olarak gerçekleştirilmesi için ön şart olarak “devrik cümle” kullanılmayacağı kabul edilmiştir.

#### 6.2.4. Tamlamalara Ait Uygulama Örnekleri

Tamlamalar sıfat tamlamaları ve isim tamlamaları olarak iki grupta incelenmiştir.

##### 6.2.4.1 Sıfat Tamlamaları

Sıfatlar, nitelediği veya belirttiği ismin önüne gelerek onunla tamlama meydana getirir. Bu tür tamlamalara sıfat tamlaması denir. Sıfat tamlamalarında sıfat tamlayan, isim tamlanandır.

Örneğin;

Güzel      ördek      →      güzel(sf[sf]) ördek(i[i])

Tamlayan      tamlanan

Uzun boylu      adam      →      uzun(sf[sf]) boy(i)\*lu(SYE[sf]) adam(i[i])

Tamlayan      tamlanan      SYE: Sıfat Yapım Eki

##### 6.2.4.2. İsim Tamlamaları

İsim tamlamaları dörde ayrılır.

###### Belirtili İsim Tamlaması

Bir varlığın kime veya neye ait olduğunu bildiren isim tamlamalarına belirtili isim tamlaması denir. Belirtili isim tamlamalarında hem tamlayan, hem de tamlanan ek alır. Tamlayan isim –in (ın, un, ün), tamlanan isim –i (ı,u,ü) ekini alır, her iki isim de sesli uyumuna uyar. Örneğin;

Yazmanın      yararı      →      yaz(f)\*ma(iYE[i])\*n(iet2)\*ın(tm) yarar(i[i])\*ı(iet3)

Tamlayan      Tamlanan

###### Belirtisiz İsim Tamlaması

Bir varlığın cinsini, ne işe yaradığını, nerede kullanıldığını bildiren isim tamlamalarına belirtisiz isim tamlaması denir. Belirtisiz isim tamlamasında tamlayan ek almaz, tamlanan isim tamlama eki olan (-i) alır.

Örneğin;

Boyama      kitabı      →      boy(i)\*a(FYE)\*ma(iYE[i]) kitap(i[i])\*ı(iet3)

Tamlayan      Tamlanan

### Takısız İsim Tamlaması

Tamlayanın, tamlananın neden yapıldığını gösterdiği isim tamlamalarına takısız isim tamlaması denir. Bu tür tamlamalarda her iki isim de takı almaz. Benzetmelerin en kısası ve anlamca en güçlüsüdür. Örneğin;

Mermer      heykel      → mermer(i[i]) heykel (i[i])

Tamlayan      Tamlanan

### Zincir İsim Tamlaması

Tamlayan ve tamlanan durumunda ikiden fazla ismin kurduğu tamlamalara zincirleme isim tamlaması denir. Zincirleme isim tamlamasında tamlayan ve tamlananın her ikisi de birer isim takımı olabilir. Örneğin;

Kadının      kız kardeşi      → kadın(i[i])\*ın(tm) kız(i[i]) kardeş(i[i])\*i(iet3)

Tamlayan      Tamlanan

## 6.3. Cümlelerin Temel ve Yan Cümleciklere Ayrılması

Sistemde kelime ayrıştırıcıdan ek ve köklerine ayrılmış olarak gelen kelimeler ve ait oldukları cümleler, öğelerine ayrılma işlemi için eğer gerekliyse parçalanırlar. Parçalama işlemi birden fazla yargı içeren cümlelere uygulanır. Cümle, temel ve yan cümlelere ayrılarak bu cümleler arasındaki bağlantı türleri belirlenir. Bu işlemler “cumle ayrıştırıcı” modülünde gerçekleştirilmektedir.

Cumle ayrıştırıcı modülünde cümleler yapı bakımından incelenir.

### 6.3.1. Yapı Bakımından Cümleler

Cümleler, bildirdikleri yargı sayısına ve öğelerin yüklemle olan ilişkisine göre çeşitlere ayrılırlar. Cümlede bir ya da birden fazla yargı olabilir. Başka bir deyişle birden fazla cümle bir araya gelip bir cümleymiş gibi görünebilir.

*Bir ceylan gibi ürktü.* → Tek yargı

*Sevincinden ne yapacağını şaşırmişti.* → İki yargı

Bu tür cümlelerde bazı ögeler ortak olduğu gibi, ögelerin tamamı farklı da olabilir. Bu cümleler birbirlerine bazı bağlaçlar yardımıyla bağlanabildiği gibi, anlam bakımından da bağlanabilirler.

*Saatine baktı ve otobüsü kaçırdığını anladı. (ve bağlacı ile bağlı)*

Cümleler yapı bakımından çeşitlere ayrılırken, içlerindeki kelime sayısı değil yüklem veya yargı sayısı dikkate alınır.

Yapı bakımından cümleler; basit, birleşik, bağlı ve sıralı olmak üzere dörde ayrılır

#### **6.3.1.1. Basit Cümle**

İçerisinde tek yargı, tek fiil, dolayısıyla isim veya fiil cinsinden tek yüklem bulunan cümledir. Başka bir cümleye bağlanmaz, yani bağımsız bir cümledir. Tamamladığı ya da onu tamamlayan bir cümlecik yoktur.

*Örnek: Zayıf kolları kirli tunç rengindeydi.*

#### **6.3.1.2. Birleşik Cümle**

Bir temel cümle ile onun anlamını tamamlayan en az bir yan cümlecikten meydana gelen cümlelerdir. Yani yapısında birden fazla cümle bulunduran cümlelerdir. Temel cümleyle yan cümlelerin bir araya geliş şekillerine göre birleşik cümleler çeşitlere ayrılır.

#### **Girişik Bileşik Cümle**

Bu tür cümlelerde yan cümlecik temel cümlecğin herhangi bir ögesi olabildiği gibi, bir ögenin parçası da olabilir. Girişik birleşik cümleler, fiilimsilerle ve çekimli fiillerle kurulur.

*Havaların ısınması / tatil düşünlerini sevindirdi. → Özne*

*Çadırları çalanlar / bulunamadı. → Sözde özne*

*Evlerin ne zaman biteceğini / bilmiyoruz. → Nesne*

*Yarın / bir tanıdığa / gideceğiz. → Dolaylı tümleç*

*Babasını karşısında görünce / çok sevindi. → Zarf tümleci*



### İç İçe Birleşik Cümle

Bir temel cümleyle, herhangi bir sebeple onun içinde kullanılan bir yardımcı cümleden oluşan cümlelerdir. Yardımcı cümle de temel cümle gibi bağımsız bir cümle yapısındadır.

Asıl yargı sonda bulunur.

Yardımcı cümle nesne olarak kullanılabilir. Alıntı hâlinededir.

*Adam, “Kartınız geçerli değil.” demez mi?*

*Şark için “Ölümün sırrına sahiptir.” derler.*

### Şartlı Birleşik Cümle

Bir temel cümle ve onun şartı olan bir cümleden oluşan birleşik cümlelerdir. Şart cümlesi tek başına yargı bildirmez; ana cümleyi zaman, şart, sebep ve benzetme yönlerinden tamamlar. Onun zarfı olarak kullanılır.

*Eğer gemi varsa / Rize’ye gemi ile gideriz.*

### 6.3.1.3. Sıralı Cümleler

Bağımsız cümlelerin, aralarındaki anlam ilgisinden dolayı virgülle veya noktalı virgülle birbiri ardına sıralanmasıyla oluşan cümleler topluluğudur. En az iki cümleden oluşur.

*Sarı çiçeğin saçları yolunmuş, kana bulanmıştı.*

*Bu, asırlardan beri böyle olagelmışti, asırlarca da böyle dürüp gidecekti.*

Sıralı cümlelerin bütün öğeleri ayrı olabildiği gibi bazıları ortak da olabilir.

*Mart kapıdan baktırır; kazma kürek yaktırır.* Özne ortak (Mart).

*Mallarımızı önce çaldılar, sonra geri bize sattılar.* Özne (Onlar) ve nesne (Mallarımızı) ortak.

*Merdivenleri kardeşin yıkasın, sen de sil.* Nesne (Merdivenleri) ortak.

*İnatçı adama dil döküyor, sürekli yalvarıyordu.* Özne (O) ve dolaylı tümleş (İnatçı adama) ortak.

#### 6.3.1.4. Bağlı Cümleler

Aralarındaki ilgiden dolayı birbirlerine bir “bağlaçla” bağlanan cümlelerdir. Bağlaçlar cümle ögesi değildir. İkiye ayrılır.

#### “ki”li Bağlı Cümleler

Farsça “ki” bağlacıyla birbirine bağlanan bağımsız cümlelerden oluşur. Yan cümle, ana cümleyi genellikle nesne ve zarf göreviyle tamamlar. Temel cümle başta, yan cümle sonda bulunur. Bu sıralanış, Türkçe cümle yapısına aykırıdır.

*Kızıl havaları seyret ki akşam olmakta.*

#### Diğer Bağlaçlarla Kurulan Cümleler

“ve, veya, ya da, fakat, ama, lâkin, hâlbuki, ne.....ne, meğer...” bağlaçlarıyla birbirine bağlanan bağımsız cümleler topluluğudur.

*Okumayı bilmiyor veya numara yapıyor.*

*Ben saatinde gelmiştim; ama o henüz ortalıkta yoktu.*

#### 6.3.2. Cümle Ayırıştırıcı Modülün İşleyişi

Sistemde varolan “ayırma kelimeleri\_ekleri” kümesine göre cümleler basit cümlelere parçalanırlar. Bu kümede fiilimsi ekleri ve bağlaçlar bulunur. Parçalanmış basit cümleler birbirleriyle “no/1” ve “no/2” şeklinde ilişkilendirilirler ve bağlantıların kaydedildiği dosyada bu numaralandırılmış cümlelerin bağlantı türleri de belirtilir. Örneğin bu bağlantı dosyasında aşağıdaki örneğe dair “1:1 ile 1:2 birbirine zarf tümleci ile bağlıdır” denilebilir. Bu durumda, “yan cümle, temel cümle zarf tümlecidir.” anlamı çıkarılır.

Örnek :

Cümle : *zeki okula giderken / beni gördü.*

Çıktı : *zeki(i[i]), okul(i[i])\*a(yon) git(f[ff])\*e(ys)\*r(gez)\*ken(ken).1:1*

*ben(zm[zm])\*i(bli) gör(f[ff])\*dü(dgz).1:2*

Bu birleşik cümle “zeki okula giderken” ve “beni gördü” şeklinde iki basit cümleye ayrılmıştır. Ancak bu cümlelerin aslında aynı cümle olduğunu anlayabilmek için her

ikisine de aynı numara olan “1” verilmiştir. 1:1, bir no’lu cümlelerin ilk basit cümlesi; 1:2 ise, bir no’lu cümlelerin ikinci basit cümlesi anlamına gelmektedir.

Cümlelerin en başındaki kelime, “ayırma kelimeleri\_ekleri” kümesinin bir elemanıysa cümlelerin ilk kelimesi silinmekte ve önceki cümleyle bağlantı türü baştaki kelimeye bağlı olarak kaydedilmektedir. Örneğin;

*Emre okula gitmedi. Çünkü bugün hastaydı. → Giriş Cümlesi*

Çıkış:

*Emre okula gitmedi. 5*

*Bugün hastaydı. 6*

*5 ile 6 sebeple bağlıdır.*

İfadeleri oluşturulmaktadır.

Bunun öncesinde cümledeki her bir kelime, kelime ayrıştırıcıdan geçirilerek ek ve köklerine ayrılmıştır.

Burda yer alan sembollerin anlamları kısaltmalar listesinde verilmektedir.

Cümle ayrıştırıcı modülünün işleyişi Tablo 6.1’de gösterilmektedir.

Tablo 6.1. Cümle Ayrıştırıcı Modülünün İşleyişi <sup>[6]</sup>

Kelimenin Yapısı	Sonraki Kelime	Bağlantı Türü
Çekim Fiil	Ve, veya, ya da, pekiştirme bağ., karşıtlık bağ, gerekçe bağ	Bağlacın kendisi
Çekim Fiil	Oysa, halbuki, rağmen	Rağmen
Çekim Fiil	Gibi	Gibi
Çekim Fiil	Diye	Amaç
Çekim Fiil	Ancak	İkinci, birincinin şartı
Çekim Fiil	Mi+”soru değilse”	Sebepl
Çekim Fiil	iken, ken	Zarf tümleci
Çekim Fiil	Çekimli fiil sondaysa	Ayırma
İsimfiil+iyelik+hal eki	Pekiştirme bağlacı	Hal ekine göre

İsimfiil+iyelik+hal eki	De	Hal ekine göre
İsimfiil+iyelik+(yön) hal eki	Kadar	Zarf tümleci
İsimfiil+iyelik+(yön) hal eki	Göre	Göre
İsimfiil+iyelik+(yön) hal eki	Karşın, rağmen	Rağmen
İsimfiil+iyelik+(yön) hal eki	doğru	Zarf tümleci
İsimfiil+iyelik+(ayr) hal eki	Dolayı, ötürü	Sebep
İsimfiil+iyelik+(ayr) hal eki	Beri	Başlama zamanı
İsim fiil+iyelik+hal eki	Hiç biri değilse	Hal ekine göre
Sıfat fiil+iyelik+hal eki	Pekiştirme bağ.	Hal ekine göre
Sıfat fiil+iyelik+hal eki	De	Hal ekine göre
Sıfat fiil+iyelik+(yön) hal eki	Karşın, rağmen	Rağmen
Sıfat fiil+iyelik+(ayr) hal eki	Dolayı, ötürü	Sebep
Sıfat fiil+iyelik+(ayr) hal eki	Beri	Başlama zamanı
Sıfat fiil+iyelik+hal eki	Hiçbiri değilse	Hal ekine göre
İsim fiil+iyelik	İle	Zarf tümleci
İsim fiil+iyelik	Karşıtlık bağ.	Bağlacın kendisi
İsim fiil+iyelik	Pekiştirme bağ.	Özne
İsim fiil+iyelik	De	Özne
İsim fiil+iyelik	İse	Özne
İsim fiil+iyelik	Gibi	Gibi
İsim fiil+iyelik	Kadar	Gibi
İsim fiil+iyelik	İçin	Amaç
İsim fiil+iyelik	Ancak	Özne
İsim fiil+iyelik	Hiç biri değilse	Özne
Sıfat fiil+iyelik	İle	Zarf tümleci
Sıfat fiil+iyelik	Karşıtlık bağ.	Bağlacın kendisi

Sıfat fiil+iyelik	Gibi	Gibi
Sıfat fiil+iyelik	Kadar	Gibi
Sıfat fiil+iyelik	İçin	Amaç
Sıfat fiil+iyelik	Pekiştirme bağ.	Bağlacın kendisi
Sıfat fiil+iyelik	Hiç biri değilse	Sıfat
İsim fiil+hal eki	Pekiştirme bağ.	Bağlacın kendisi
İsim fiil+hal eki	De	Hal ekine göre
İsim fiil+ (yön) hal eki	Göre	Göre
İsim fiil+ (yön) hal eki	Doğru	Zarf tümleci
İsim fiil+ (ayr) hal eki	Dolayı, ötürü	Sebep
İsim fiil+hal eki	Hiçbiri değilse	Hal ekine göre
Sıfat fiil+hal eki	Pekiştirme bağ.	Bağlacın kendisi
Sıfat fiil+hal eki	De	Hal eklerine göre
Sıfat fiil+ (yön) hal eki	Göre	Göre
Sıfat fiil+ (ayr) hal eki	Dolayı, ötürü	Sebep
Sıfat fiil+ (yön) hal eki	Kadar	Zarf tümleci
Sıfat fiil+ (yön) hal eki	Karşın, rağmen	Rağmen
Sıfat fiil+hal eki	Hiç biri değilse	Hal eklerine göre
İsim Fiil	İle	Zarf tümleci
İsim Fiil	Karşıtlık bağ.	Bağlacın kendisi
İsim Fiil	Pekiştirme bağ.	Özne
İsim Fiil	de	Özne
İsim Fiil	ise	Özne
İsim Fiil	Gibi	Gibi
İsim Fiil	Kadar	Gibi
İsim Fiil	İçin	Amaç

İsim Fiil	Üzere, üzre	Amaç
İsim Fiil	Üzere+yeterlilik	Şart
İsim Fiil	üzere	Ayırma
İsim Fiil	ancak	Özne
İsim fiil	Hiçbiri değilse	Özne
Sıfat Fiil	İçin	Özne
Sıfat Fiil	İle	Zarf tümleci
Sıfat Fiil	Karşıtlık bağ.	Bağlacın kendisi
Sıfat Fiil	De	Özne
Sıfat Fiil	Gibi	Gibi
Sıfat Fiil	Hiçbiri değilse	Sıfat
Zarf Fiil	Ne olursa olsun	Zarf fiil

#### 6.4. Basit Cümlelerin Öğelerine Ayrılması

Tüm cümleler basit cümleler şeklinde parçalandıktan sonra tek bir yüklem içerir hale getirilmiştir. Cümleler Prolog derleyicisine gönderilmeden önce öğelerine ayrılmalıdır. Bu işlemler öge ayrıştırıcı modülde gerçekleştirilmektedir.

##### 6.4.1. Cümlenin Öğeleri

Bir duyguyu, düşüncüyü, isteği, haberi, durumu, olayı v.b. ifade etmek için kurulan ve kendi içinde bir anlam ve yargı bütünlüğü olan sözcüğe veya söz dizisine “cümle” denir. Her bir cümle bir yüklem ve varsa ona bağlı diğer öğelerden oluşur. Cümlede yargıyı bildiren öge yüklemidir.

Öge, cümleyi oluşturan bölümlerin herbiridir. Her öge görev ve anlam yönünden bir tek öğeye eşlik eder ve bu öge de birinci derece önem taşıyan yüklemse öğeler çeşitli şekillerde birbirlerinden ayrılabilirler.

*Bu sabah / işe gitmek için / çok / sabırsızdı . ( Z.T. / Z.T. / Z.T. / Y.)*

Vugulanmak istenen öge yüklemine önüne getirilir.

*Bu işi/ ben /yaptım. (Belirtili Nesne / Özne / Yüklem)*

Öğeleri daha ayrıntılı inceleyecek olursak :

#### 6.4.1.1. Yüklem

İş, kılış, oluş, hareket, durum bildiren; heber veren; cümleyi bir yargıya bağlayan çekimli ögedir.

*Sabaha kadar dolaştı.* ( Fiil Cümlesi )

Fiil cümlesinin yüklemi çekimli bir fiildir. İsim cümlesinin yüklemi ise ek-fiille çekimlenmiş bir isimdir.

*O, bu dünyada treni kaçırmış olanlardan biridir.* ( İsim Cümlesi)

Yüklemi belirtilmemiş cümleler eksilteli cümlelerdir. Bu tip cümleleri tamamlamak bu projenin kapsamı dışındadır.

*Kıratın yanında duran ya huyundan ya suyundan.*

#### 6.4.1.2. Özne

Yüklemde bildirilen işi, oluşu, hareketi, durumu, kılışı yerine getiren; hakkında bilgi ve haber veren ögedir. Yani yaparı veya olanı belirtir.

*Karşıdan gelen kara şovalyeler etrafa korku saçıyorlardı.*

Özne, yükleme sorulan “ne / kim” sorularına yanıt verir. Özne olan kelime ya da kelime grupları cümlede hiç bir hal eki almadan kullanılırlar. Ama çoğul ya da iyelik eki alabilirler. Proje de bu özellikler göz önünde bulundurulmuştur. Özne, anlamdan çıkarılabileceği ya da tekrardan dolayı anlatım bozukluğu yaratabileceği için söylenmeyebilir. Özne söylenmediğinde “gizli özne” adını alır. Gizli özne yüklem taşıdığı şahıs ekinden anlaşılır.

*Korkularına yenik düşmüş(ler)di. / \* Onlar : Gizli Özne \*/*

#### Özne – Yüklem Uyumu

Özne ve yüklem “olumluluk – olumsuzluk” ve “tekillik – çoğulluk” yönlerinden uyum göstermelidir.

i. *Olumluluk – Olumsuzluk Uyumu*

Özne olumlu ise yüklem de olumlu; özne olumsuz ise yüklem de olumsuzdur.

*Herkes yüreğini ortaya koyarak sahaya çıktı.*

ii. *Tekillik – Çoğulluk Uyumu*

Özne tekilse yüklem de tekil; özne çoğulsa yüklem de çoğul olur. .

Kötüler elleri boş olarak dönecekler.

**6.4.1.3. Nesne**

Yüklemde bildirilen ve öznenin yaptığı işten doğrudan etkilenen öge, nesnedir. Dolayısıyla fiil cümlelerinden sadece yüklemi geçişli fiil olanlar, nesne alır. Pek rastlanmasa da isim cümleleri de nesne alabilir.

Özne olabilen bütün sözcükler, kelime grupları ve iç cümleler nesne olabilir. Yükleme sorulan “ *ne, neyi, kimi* ” sorularına yanıt verir.

*Kibritçi kız, esen rüzgarda bir kelebek misali kanatlarını çırpıyordu.* (Neyi?)

Nesneler belirtili ve belirtisiz olmak üzere ikiye ayrılır. Belirtme hal eki alanlar belirtili; yalın halde bulunanlar da belirtisiz nesnelerdir.

*Çok güzel gitar çalardı.* (Ne?–Belirtisiz Nesne)

*Gitari çok güzel çalıyordu.* (Neyi?–Belirtili Nesne)

Belirtisiz nesne daima yüklemden önce gelir. Yüklemle belirtisiz nesne arasında “de, dahi, bile” edatlarından başka bir kelime giremez.

Önüne gelen, korkunç hikayeler anlatıyordu. (Belirtisiz Nesne)

**6.4.1.4. Dolaylı Tümleç**

“-e, -de, -den” eklerini alarak cümlemin, dolayısıyla yüklemnin anlamını, “fiilin, ayrılma, bulunma ve yönelme bakımlarından ilgili olduğu” yer yönünden tamamlayan ögedir.

*Halk, saldırılara karşı meydanda toplanmıştı.* (bulunma, yer)

Yükleme “*nereye, nerede, nereden, kime, kimden, neye, neyde, neyden*” sorularının cevabıdır.

*Bunları bir bilene danışmalısın.* (Kime?)

*Evler ağaçtan yapıldığından kolay ve hızlı bir şekilde yanmıştı.* (Neyden?)



#### 6.4.1.5. Zarf Tümlenci

Yüklemin anlamını zaman, durum, yön, miktar, tarz, vasıta, şart, sebep, birliktelik yönlerinden tamamlayan kelime ya da kelime gruplarıdır. Edat tümlenci olarak tanımlanan tümleçler de birer zarf tümlencidir.

*Sabaha kadar zaferi kutladılar.*

*Yaşadığı mutluluğu anlatırken yerinde duramıyordu.*

Yükleme “nasıl, ne zaman, ne kadar, nereye, kiminle, neyle, niçin, neyden, niye” soruları sorularak bulunur.

*Komutan büyük bir öfkeyle içeriye girdi. (Nasıl?, Nereye?)*

*O küçük kız, geceleri, yağmur taneleri cama vururken korkuyla uyumaya çalışırdı.*

Ne zaman?

Ne zaman?

Nasıl?

#### 6.4.2. Basit Bir Cümleyi Öğelerine Ayıran Modülün Çalışma Adımları

Tablo 6.2’de öge ayrıştırıcı modülde, basit bir cümlemin ögelerine ayrılması için izlenen yöntemlerin bir özeti verilmiştir.

Tablo 6.2. Öğelerine Ayırma Kuralları

Ögenin Özelliği ( içeriğine göre )	Öge Çeşidi
(zf)	Özne
(i)	Özne
(s) içeriyor ve sonrası sıfat değilse	Özne
Sonunda (çe) varsa	Özne
Sonunda (tm)*ki(ki) varsa	Özne
Sonunda (ie) varsa	Özne
Sonunda (i) varsa ve sonrası (ie) içermiyorsa	Özne
Sonunda ki(ki) varsa	Sıfat (öge parçasıdır)
(bln)	Belirtili Nesne
(yon) ve sonrasında kadar varsa	Zarf Tümlenci
(yon) ve sonrasında göre varsa	Zarf Tümlenci
Sonrasında için varsa	Zarf Tümlenci
Sonrasında karşı varsa	Zarf Tümlenci
(yon) ve sonrası doğru ise	Zarf Tümlenci / Dolaylı Tümlenç
(ayr) içeriyorsa ve sonrasında dolayı ya da ötürü geliyorsa	Zarf Tümlenci
(ayr) ve sonrasında beri geliyorsa	Zarf Tümlenci
Sonunda (zf) varsa	Zarf Tümlenci

Sonunda ( <b>ayr,bul,yon</b> ) varsa	Dolaylı Tümleç
( <b>s</b> ). içeriyorsa	Yüklem
Zaman, şahıs, ekfiil, fiilimsi eki içeriyorsa	Yüklem
Sonrasında <b>ve, veya, ya da</b> varsa	Öge parçasıdır.
Sonrasında <b>ile</b> geliyorsa	Zarf Tümleci
Sonunda ( <b>i</b> ) varsa ve sonrası ( <b>ie</b> ) içeriyorsa	Öge parçasıdır
( <b>tm</b> ) içeriyorsa	(ie) içeren ilk kelime dahil öge parçasıdır.

### 6.5. Basit Cümlelerin ve Bağlantılarının Prolog Formatına Dönüştürülmesi

Öğelerine ayrılmış cümleler, Prolog’da kullanılabilmesi için daha basit yapıya dönüştürülür .

Tüm basit cümle yapısındaki cümleler için seçilen format :

**vardir(yüklem, özne, belirtili nesne, belirtisiz nesne, dolaylı tümleç, zarf tümleci, id, yüklem kipi, yüklem zamanı).** (6.1)

Bu şekildeki bir formatla, cümlelerin istenilen ögesine kolaylıkla erişilebilir. Cümlede kullanılmayan öğeler varsa yukarıdaki formatta yerleri boş bırakılır. Eğer cümlede birden fazla aynı öğeden varsa, örneğin birden fazla dolaylı tümleç varsa, liste yapısı kullanıldığından bir sorun yaşanmaz.

Tüm birbiriyle bağlantılı cümleler için seçilen format :

**bag(bağ\_türü, id1, id2).** (6.2)

Bu ifade; “id1” id’sine sahip cümleyle “id2” id’sine sahip cümle arasında “bağ\_türü” şeklinde bir bağlantı olduğu anlamına gelir.

Kullanıcının girdiği tüm cümleler bu iki Prolog cümlesine çevrilir. Elde edilen bu Prolog cümleleri Prolog veritabanına çalışma esnasında yerleştirilir (assert işlemi). Örneğin aşağıdaki şekilde bir paragraf girilirse:

*“Zeki İtüspor taraftarıdır. Bu hafta Egesporla İtüspor maç yapacaklar. Birçok insan İtüsporun yeneceğine inanıyor.”*

Bu durumda oluşan Prolog formu:

- ♣ `vardir(taraftar(i[i]*ı(iet3)*dır(gez),zeki(i[i]),[],itüspor(i[i]),[],[],800,'',gez).`
- ♣ `vardir(yap(f[f])*acak(gz)*lar(seç3),egespor(i[i])*la(bir)itüspor(i[i]),[],maç(i[i]),  
[], [bu(sf[sf])hafta(zf[zf])],801,'',gz).`
- ♣ `vardir(itüspor(i[i])*un(tm)ye(f)*n(FYE[f]*ecek(sfe1)*ı(iet3)*n(kh)*e(yon),  
birçok(sf[sf]) insan(i[i]),[],[],[],801:1,'',')).`
- ♣ `vardir(inan(f[f])*ı(ys)*yor(şz),'',[],[],[],801:2,'',şz).`
- ♣ `bag(dt,800:1,800:2).`

Burda dikkat edilmesi gereken, İtüspor ve Egespor gibi birtakım özel isimlerin sistemin sözlüğüne isim olarak kaydedilmiş olması gerektiğidir. Aksi takdirde bunlar özel isim olduklarından, sistem bunların isim olduğunu anlayamayabilir.

## 6.6. Soru Modülü

Prolog formatında tutulan cümlelerden soru üretirken, Prolog veritabanı kullanıcı tarafından seçilen öğeye bağlı olarak sorgulanır. Örneğin kullanıcı özne ile ilgili sorular sormak isterse kullanıcıya cevabı özne olan sorular yöneltilir. Kullanıcının aradığı öğeyi içeren Prolog cümleleri taranır ve bu cümleler soru formuna dönüştürülür. Prolog formlarını birer soru formuna dönüştürürken, Prolog'daki formda bulunan cümlede o öğe çıkartılır ve ekleri atılır. Aranan öğeyi oluşturan kelime ya da kelime grupları atılır ve yerine o öğeye ait olan soru kelimesi getirilir. Böylece cevabı daha önce atılmış öğe olan soru cümlesi oluşturulmuş olur ve kullanıcıya sorulur. Bu sırada doğru cevap da zaten bilgisayar tarafından biliniyor olur.

Zarf tümleci gibi bazı öğeler için “nasıl”, “ne zaman”, “ne kadar” gibi birden fazla soru kelimesi ihtimali vardır. Hangi soru kelimesinin kullanılacağını belirlemek üzere Prolog sorgulamasının yanısıra, o öğeyi içeren sözcüklerin ekleri de gözönünde bulundurulur. Örneğin;

*Hızlı\*ca → Nasıl ?*

*Gelin\*ce → Ne zaman ?*

*Git\*eli → Ne zamandan beri ?*

Diğer öğelerle ilgili soru üretmek için, ilgili öğenin bulunmasında yükleme yöneltilen sorulardan yararlanılmaktadır. Örneğin ismin –e halini (yönelme) almış

dolaylı tümleç ögesi için, yanıtı bu öge olan ve “Nereye” ile başlayan soru üretilerek kullanıcıya soru sorulur. Ayrıntılı bilgi alt bölümlerdedir.

### 6.6.1. Özneye Dayalı Soru Oluşturma Modülü

Cümle: *Zeki okula gitti.*

Prolog Formu: *vardir(git(f[f])\*di(dgz),zeki(i[i]),[],[],okul(i[i])\*a (yön),[],144, ‘’,dgz).*

Soru: *Kim okula gitti?*

Özne vardır formatının 2 numaralı indisindedir. O halde bu cümlede özneyi bulmak için “**vardir(,X,,,,).**” Şeklinde bir sorgulama yapmak gerekir. Bu durumda yanıt olarak zeki(i[i]) dönecektir. Oysa cevap sadece zeki ‘dir. O nedenle burdaki eklerin temizlenip zeki kelimesinin elde edilmesi gerekir. Daha sonra metinde geçen orjinal cümledeki Zeki kelimesi atılır ve yerine özne sorusu olan “Kim” ya da “Ne” getirilir. Bilindiği gibi özne özel bir isim değilse “Ne”, özel bir isimse “Kim” şeklinde sorgulanır. Bu problemi çözmek için öncelikle özel isimlerin saklı tutulduğu ‘name.txt’ dosyası taranır. Eğer atılan sözcük olan Zeki bu dosyada mevcutsa ‘Kim’, mevcut değilse ‘Ne’ soru kelimesi eklenerek soru üretilir.

### 6.6.2. Nesneye Dayalı Soru Oluşturma Modülü

Cümle: *Zeki kediye sevdi.*

Prolog Formu: *vardir(sev(f[f])\*di(dgz),zeki(i[i]),kedi(i[i])\*y (kh)\*i(bli),[],[],144, ‘’,dgz).*

Soru: *Zeki neyi sevdi?*

Bu durumda Prolog veri tabanı “**vardir(,X,Y,,,,).**” ile sorgulanır.

X: Belirtili Nesne’ye sahip cümleleri,

Y: Belirtisiz Nesne’ye sahip cümleleri bulmak için kullanılır.

Bu sorgulama sonunda X bir değerle dönerken, Y; metinde belirtisiz nesne yoksa ‘Null’ (boş) olarak döner. Yukarıdaki prolog formunun bu şekilde sorgulanmasıyla;

X: ‘kedi(i(i))\*y(kh)\*i(bli)’ ile Y: Null ile döner. Bu durumda X ifadesindeki ekler atılır ve ‘kediye’ kelimesi cevap olarak bilgisayar tarafından bulunduktan sonra, soru kelimesi üretmek amacıyla cümleden atılır ve bunun yerine belirtili nesne sorusu olan ‘Neyi’ ya da ‘Kimi’ sözcüğü getirilerek soru cümlesi üretilir. Hangisinin getirilmesi gerektiği yine ‘name.txt’ dosyasına bakılarak anlaşılır. Kedi cansız bir varlık olduğundan bu dosyada yoktur ve ‘Neyi’ sorusu kullanıcıya yöneltilir.

### 6.6.3. Dolaylı Tümlece Dayalı Soru Oluşturma Modülü

Cümle: *Zeki okula gitti.*

Prolog Formu: *vardir(git(f[f])\*di(dgz),zeki(i[i]),[],[],okul(i[i])\*a(yön),[],144, ',',dgz).*

Soru: *Zeki nereye gitti?*

Dolaylı tümlece dayalı soru üretmek için prolog veri tabanı “**vardir**(\_,\_,\_,**X**,\_,\_,\_)” ifadesiyle sorgulanır. Dolaylı tümleç beşinci indistedir. Bu sorgulama sonucunda X, ‘okul(i(i))\*a(yon)’ ile geri döner ve ekler ayıklanarak doğru cevabın olduğu ‘okula’ sözcüğü elde edilir. Okula sözcüğü orjinal cümleden atılır ve yerine uygun soru kelimesi koyularak soru sorulur.

Dolaylı tümleç soru kelimeleri:

- ▲ Kim, Kimde, Kimden;
- ▲ Neye, Neyde, Neyden;
- ▲ Nereye, Nerede ve Nereden ‘dir.

Bunlardan hangisinin doğru soru kelimesi olduğunu saptamak için öncelikle ‘name.txt’ dosyasının aranan kelime olan okula sözcüğünü içerip içermediğine bakılır. Eğer içeriyorsa ‘Kim’ sözcüğü belirlenir. Eğer içermiyorsa, object.txt dosyasına bakılır. Eğer kelime bu dosyada mevcutsa, soru kelimesi olarak ‘Ne’ seçilir. Eğer kelime her iki dosyada da yoksa soru kelimesi ‘Nere’ olarak belirlenir.

İkinci işlem soru kelimesine getirecek olan ekin tespit edilmesidir. Bunun için sözcüğün eklerine bakılır. Örneğin kelimenin eklerinde (yön) geçiyorsa bu yönelme eki olan “-e,-a” içeriyor demektir ve soru kelimesinin sonuna ‘-e’ eklenir. Eğer eklerinde (ayr) içeriyorsa soru kelimesinin sonuna ‘-den’; eğer eklerinde (bul) içeriyorsa soru kelimesinin sonuna bulunma eki olan “-de” getirilir. Örnekteki cümle için soru kelimesi ‘Nereye’ dir.

### 6.6.4. Zarf Tümlecine Dayalı Soru Oluşturma Modülü

Cümle : *Zeki sabahleyin okula gitti.*

Prolog Formu:

*vardir(git(f[f]) \*di(dgz),zeki(i[i]),[],[],okul(i[i])\*a(yon),[sabahleyin],144, ',',dgz).*

Soru: *Zeki ne zaman okula gitti?*

Zarf tümleci soruları üretirken Prolog veritabanı “**vardir**(\_,\_,\_,\_,**X**,\_,\_)” ile sorgulanır. Zarf tümleci vardır yapısının altıncı indisidir. Sorgulamanın sonucunda X, ‘sabahleyin’ değeri ile geri döner. ‘Sabahleyin’ sözcüğü orjinal cümleden atılır ve bunun yerine soru kelimesi eklenir. Zarf tümleci için soru kelimesi belirlenirken üç olasılık mevcuttur:

- ▲ Zaman bildiren zarflar için → Ne Zaman?
- ▲ Miktar bildiren zarflar için → Ne Kadar?
- ▲ Durum bildiren zarflar için → Nasıl?

Sorgulama sonucunda elde edilen zarf tümleci ögesi öncelikle miktar bildiren zarfları içeren dosyada aranır. Eğer burda yoksa, zaman zarflarını içeren dosyada aranır. Eğer bu dosyalardan birindeyse ilgili soru kelimesi bulunmuştur. Eğer her iki dosyada da bu kelimeye rastlanmamışsa, çok geniş bir kümeye sahip olan ve tamlama yapılarında da bulunabilen ‘durum zarfları’ sınıfına ait soru kelimesi olan ‘Nasıl’ sözcüğü, soru kelimesi olarak saptanır. Örnek cümle için soru kelimesi ‘Ne zaman’ dır. Çünkü ‘sabahleyin’ sözcüğü zaman bildiren zarflar dosyasında bulunmaktadır.

## 6.7. Yanıt Kontrol ve Doğrulama Modülü

Sorular üretilirken cevaplar da sistem tarafından tutulur. Bu işlem iki şekilde gerçekleşir:

- a. Kullanıcının girebileceği tek kelimelik yanıt olasılıklarına karşılık, direkt sorgulanan ögenin kendisinin saklanması,
- b. Cevabı içeren cümlenin tamamının metin içindeki yerini gösteren “id” numarasının saklanması.

Kullanıcı yanıtı girdiğinde yanıt öncelikle ilk yönteme göre karşılaştırılır. Eğer doğru eşleşme varsa kullanıcıya ‘Verdiğiniz yanıt doğrudur’ mesajı iletilir. Aksi halde kullanıcı uyarılır ve doğru yanıt iletilir.

Eğer kullanıcı bir kaç kelimelik bir yanıt verdiyse, öncelikle bu yanıt önceki bölümlerde belirtilen adımlardan geçirilerek öğelerine ayrılır ve Prolog formuna dönüştürülür. Cevabın saklı olduğu Prolog dosyasından seçilen sorunun id’si okunur. ve doğru yanıtın Prolog önermesiyle, kullanıcının verdiği yanıtın Prolog önermesi

karşılaştırılır. Bu karşılaştırma işlemi sadece belirli kurallar çerçevesinde yapılmaktadır. Karşılaştırma işlemi sadece yüklem ve sorulan ögenin her iki formda da karşılaştırılması mantığına dayanır. Eğer karşılaştırma sonucu hata ile dönerse, kullanıcıya “Verdiğiniz yanıt yanlıştır. Doğru yanıt aşağıda bulunan kutucuktaki gibidir” şeklinde bir mesaj iletilir. Daha sonra doğru yanıt bu kutucukta belirir.

Hassasiyet arttırılmak istenirse, cevabı içeren ve metinde geçen cümlelerin id’si okunarak, metindeki cümle bulunur ve kullanıcının verdiği cevapla birebir karşılaştırma yapılabilir.

## 7. ARAYÜZ TASARIMI VE EKRAN GÖRÜNTÜLERİ

### 7.1. Örnek Uygulama I

#### Girilen Metin (1):

“Ali otobüsle Amerika’ya gitti. Otobüste yaşlı bir bayanla tanıştı. Otobüsten gelirken ayağını kırdı. Bu yüzden hastahaneye kaldırıldı. Hastahane ücretini ödeyemediğinden mahsur kaldı.”

Metnin ISAPI ile çözümlenmesi (dll ile internet üzerinden analiz):

Programın ilk aşaması Internet üzerinden ISAPI teknolojisi kullanılarak gerçekleştirilmektedir. Şekil 7.1’de metinle ilgili ilk giriş ekranı verilmektedir.



Metin İşleme: Soru Soran Bir Sistem Tasarımı

**Türkçe Cümle Analizi**

**...Metni Giriniz...**

Ali otobüsle amerikaya gitti. Otobüste yaşlı bir bayanla tanıştı. Otobüsten inerken ayağını kırdı. Bu yüzden hastahaneye kaldırıldı Hastahane ücretini ödeyemediğinden mahsur kaldı.

**Analiz Et**

**Sil**

Şekil 7.1. Giriş Ekranı

Şekil 7.2’de girilen metnin her bir kelimesi ek ve köklerine ayrılmakta, birleşik cümleler basit cümle yapısına dönüştürülmektedir.



Şekil 7.3.'te ise girilen metnin basit cümlelere dönüştürülmüş halinin, her bir cümle için öğelerine ayrılması gösterilmektedir.

### Metindeki Cümlelerin Kök ve Eklerine Ayrılmış Sekli

```
ali(i[i]) otobüs(i[i])*le(birl) amerika(i[i])*y(kh)*a(yon) git(f[f])*di(dgz).651 otobüs(i[i])*de(bul) yağlı(sf[sf]) bir(sf[sf]) bayan(i[i])*la(birl) tanış(i[i])*dı(ef_dgz).652 otobüs(i[i])*den(ayr) gel(f[f])*i(ys)*r(gez)*ken(ken) ayağ(i[i])*ı(iet3)*n(kh)*ı(bli) kır(i[i])*dı(ef_dgz).653 bu(sf[sf]) yüzden(zf[zf]) hastahane(i[i])*y(kh)*e(yon) kaldırıl(f[f])*dı(dgz) hastahane(i[i]) ücret(i[i])*ı(iet3)*n(kh)*ı(bli) öde(f[f])*y(kh)*e(ytk)*me(ol)*dik(sfe2)*ı(iet3)*n(yu)*den(ay) mahsur(sf[sf]) kal(i[i])*dı(ef_dgz).654
```

### ...Metindeki Cümlelerin Basit Cümle Yapısına Dönüştürülmüş Şekli...

```
ali(i[i]) otobüs(i[i])*le(birl) amerika(i[i])*y(kh)*a(yon) git(f[f])*di(dgz).651 otobüs(i[i])*de(bul) yağlı(sf[sf]) bir(sf[sf]) bayan(i[i])*la(birl) tanış(i[i])*dı(ef_dgz).652 otobüs(i[i])*den(ayr) gel(f[f])*ı(ys)*r(gez)*ken(ken).653:1 ayağ(i[i])*ı(iet3)*n(kh)*ı(bli) kır(i[i])*dı(ef_dgz).653:2 bu(sf[sf]) yüzden(zf[zf]) hastahane(i[i])*y(kh)*e(yon) kaldırıl(f[f])*dı(dgz).654:1 hastahane(i[i]) ücret(i[i])*ı(iet3)*n(kh)*ı(bli) öde(f[f])*y(kh)*e(ytk)*me(ol).654:2:1 mahsur(sf[sf]) kal(i[i])*dı(ef_dgz).654:2:2
```

Cümleleri Öğelerine Ayırmak İçin Aşağıdaki Butona Tıklayınız

**Öğelerine Ayır**

Şekil 7.2. Metnin Ek ve Köklerine Ayrıldığı ve Herbir Bileşik Cümlelerin Basit Cümle Yapısına Dönüştürüldüğü Ekran

Cümlelerin Ögelerine Ayrılmış Şekli Şöyledir:

ali(i[i]) ----> ö / otobüs(i[i])\*le(bir1) amerika(i[i])\*y(kh)\*a(yon) ----> dt / git(f[f])\*dı(dgz). ----> yük /  
otobüs(i[i])\*de(bul) ----> dt / yaşh(sf[sf]) bir(sf[sf]) bayan(i[i])\*la(bir1) tanış(i[i])\*dı(ef\_dgz). ----> yük /  
otobüs(i[i])\*den(ayr) ----> dt / gel(f[f])\*ı(ys)\*r(gez)\*ken(ken). ----> yük /  
ayağ(i[i])\*ı(iet3)\*n(kh)\*ı(bli) ----> bln / kır(i[i])\*dı(ef\_dgz). ----> yük /  
bu(sf[sf]) yüzden(zf[zf]) ----> zt / hastahane(i[i])\*y(kh)\*e(yon) ----> dt / kaldırıl(f[f])\*dı(dgz). ----> yük /  
hastahane(i[i]) ücret(i[i])\*ı(iet3)\*n(kh)\*ı(bli) ----> bln / öde(f[f])\*y(kh)\*e(ytk)\*me(ol). ----> yük /  
mahsur(sf[sf]) ----> zt / kal(i[i])\*dı(ef\_dgz). ----> yük /

ilk Sayfaya Dön

Şekil 7.3. Girilen metnin Ögelerine Ayrılışı Ekran

### Programın Girilen Metinle ilgili Ögelere Göre Gruplandırılmış Soruları:

#### Nesne Soruları :

*Otobüsten gelirken neyi kırdı?*

*Neyi ödeyemediğinden mahsur kaldı?*

#### Zarf Tümleci Soruları :

*Nasıl hastahaneye kaldırıldı?*

*Hastahane ücretini ödeyemediğinden nasıl kalırıldı?*

*Ne zaman ayağını kırdı?*

#### Dolaylı Tümleç Soruları :

*Ali nereye otobüsle gitti?*

*Nerede yaşlı bir bayanla tanıştı?*

*Nereden gelirken ayağını kırdı?*

*Bu yüzden nereye kaldırıldı?*

#### Özne Soruları :

*Kim Amerika'ya otobüsle gitti?*

*Kim otobüste yaşlı bir bayanla tanıştı?*

*Kim otobüsten gelirken ayağını kırdı?*

*Kim bu yüzen hastahaneye kaldırıldı?*

*Kim hastahane ücretini ödeyemediğinden mahsur kaldı?*

## Metne Göre Delphi Ekran Görüntüleri:

Programın ikinci bölümü olan Dephi diliyle yazılmış ve metinle ilgili kullanıcıya soruların yöneltildiği kısım Şekil 7.4 ve Şekil 7.5'te gösterilmektedir.

**Türkçe Metin İşleme: Soru Soran Bir Sistem Uygulaması**

Metin İşleme: Soru Soran Bir Sistem Uygulaması (SSS)

...Girdiğiniz Metin...

ali otobüsle amerikaya gitti.  
otobüste yaşı bir bayanla tanıştı.  
otobüsten gelirken ayağını kırıdı.  
bu yüzden hastahaneye kaldırıldı.  
hastahane ücretini ödeyemediğinden mahsur kaldı.

Hangi Öğe Yönelik Soru Sorulsun ?

Özne  
Dolaylı Tamlık  
Zarf Tamlacı  
Nesne

Yanıtı Kontrol Et

Metni Sil

ÇIKIŞ

...Seçilen öge ile ilgili metinden üretilen sorular...

otobüsten gelirken neyi kırıdı.  
neyi ödeyemediğinden mahsur kaldı.

Cevabınızı aşağıdaki kutucuğa yazınız :

parayı

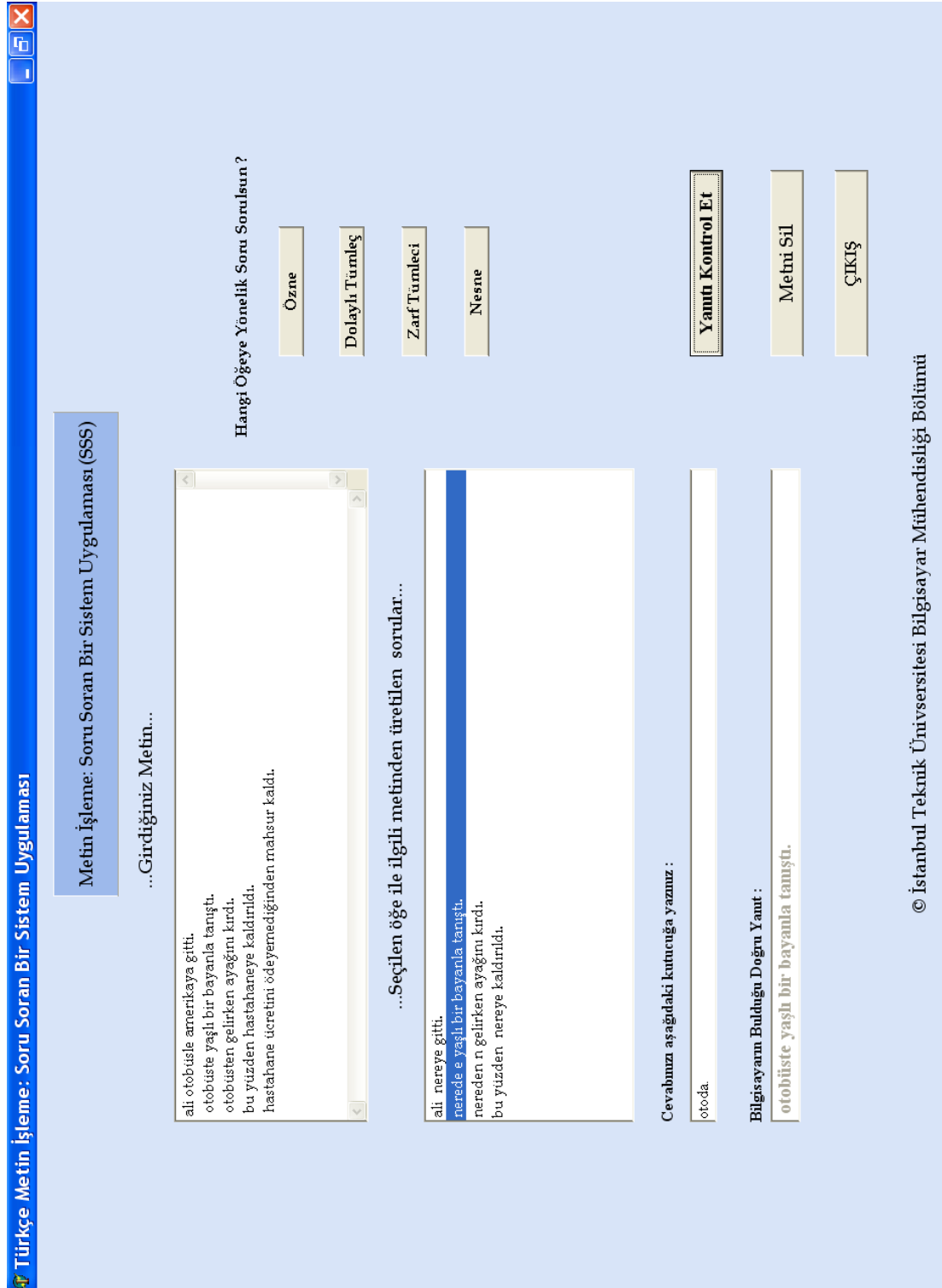
Bilgisayarı Bulduğunuz Doğru Yanıt :

hastahane ücretini

© İstanbul Teknik Üniversitesi Bilgisayar Mühendisliği Bölümü

Şekil 7.4. Metinle İlgili Nesne Sorularının Çıkarıldığı Ekran

Şekil 7.5'te ise dolaylı tümleci buldurmaya yönelik soruların gösterildiği arayüzler gösterilmektedir.



Türkçe Metin İşleme: Soru Soran Bir Sistem Uygulaması

Metin İşleme: Soru Soran Bir Sistem Uygulaması (SSS)

...Girdiğiniz Metin...

Hangi Öğeyle Yönelik Soru Sorulsun ?

Özne

Dolaylı Tümleş

Zarf Tümleci

Nesne

...Seçilen öge ile ilgili metinden üretilen sorular...

ali nereye gitti.  
nereye e yaşlı bir bayanla tanıştı.  
nereden n gelirken ayağını kırıdı.  
bu yüzden nereye kaldırıldı.

Yanıtı Kontrol Et

Metni Sil

ÇIKIŞ

Cevabınızı aşağıdaki kutucuğa yazınız :

otoda

Bilgisayarın Bulduğu Doğru Yanıt :

otobüste yaşlı bir bayanla tanıştı.

© İstanbul Teknik Üniversitesi Bilgisayar Mühendisliği Bölümü

Şekil 7.5. Dolaylı Tümleş Sorularının Çıkarıldığı Ekran

## 7.2. Örnek Uygulama II

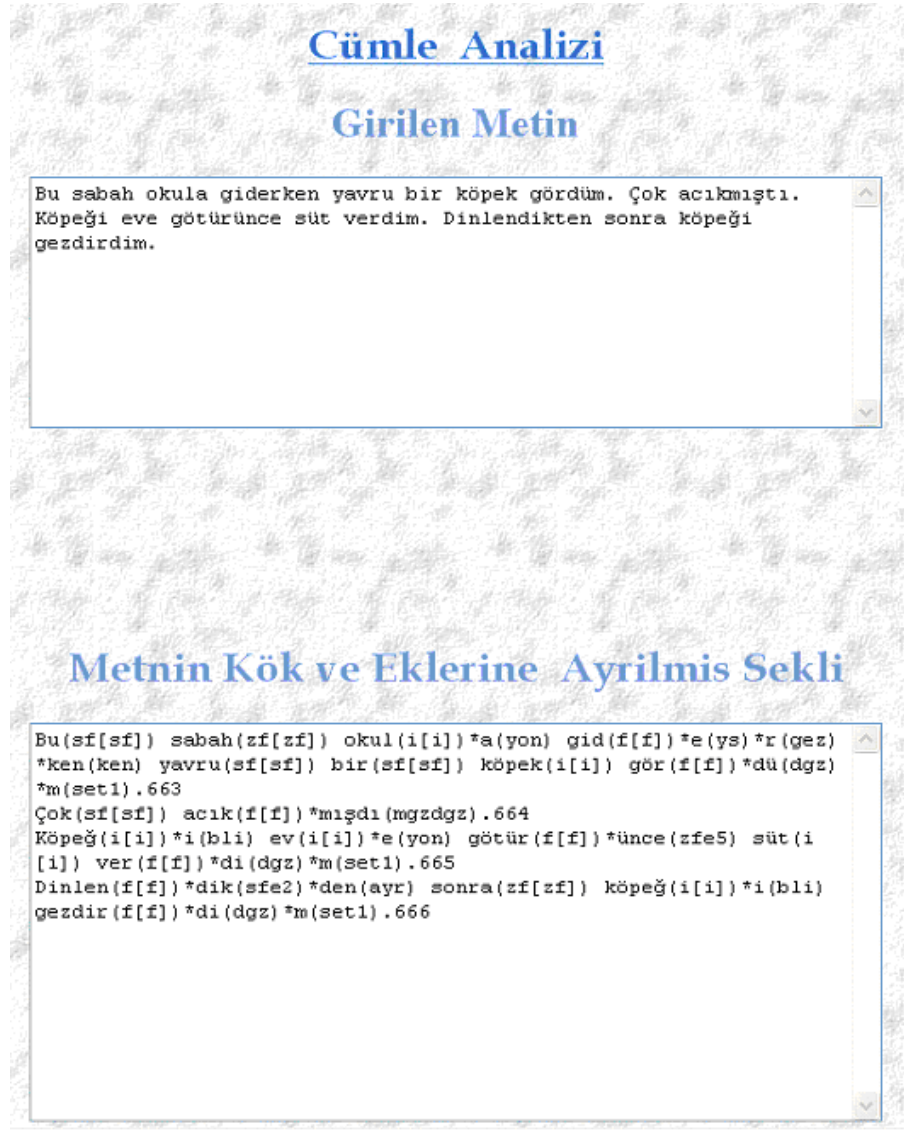
### Girilen Metin (2):

“Bu sabah okula giderken yavru bir köpek gördüm. Çok acıkmıştı. Köpeği eve götürünce süt verdim. Dinlendikten sonra köpeği gezdirdim.”

Metnin ISAPI ile çözümlenmesi (dll ile internet üzerinden analiz):

Şekil 7.6’da girilen ikinci metnin Internet üzerinden çözümlendiği ekran görüntüsü verilmektedir.

Şekil 7.7’de girilen metin ek ve köklerine ayrıldıktan sonra bileşik cümleler, basit cümlelere dönüştürülmektedir.



Şekil 7.6. Metnin internet üzerinden çözümlendiği ekran



## Metindeki Cümlelerin Basit Cümle Yapısına Dönüştürülmüş Sekli

```
Bu(sf[sf]) sabah(zf[zf]) okul(i[i]) *a(yon) gid(f[f]) *e(ys)*r(gez)
*ken(ken).663:1
yavru(sf[sf]) bir(sf[sf]) köpek(i[i]) gör(f[f]) *dü(dgz)*m
(set1).663:2
Çok(sf[sf]) acık(f[f]) *mışdı(mgzdgz).664
Köpeğ(i[i]) *i(bli) ev(i[i]) *e(yon) götür(f[f]).665:1
süt(i[i]) ver(f[f]) *di(dgz)*m(set1).665:2
Dinlen(f[f]).666:1
sonra(zf[zf]) köpeğ(i[i]) *i(bli) gezdir(f[f]) *di(dgz)*m
(set1).666:2
```

Cümlelerin Öğelerine Ayrılmış Sekli İçin Aşağıdaki Butona Tıklayınız

Ögelerini Göster

Şekil 7.7. Girilen Metnin Basit Cümleler Haline Getirilmesi

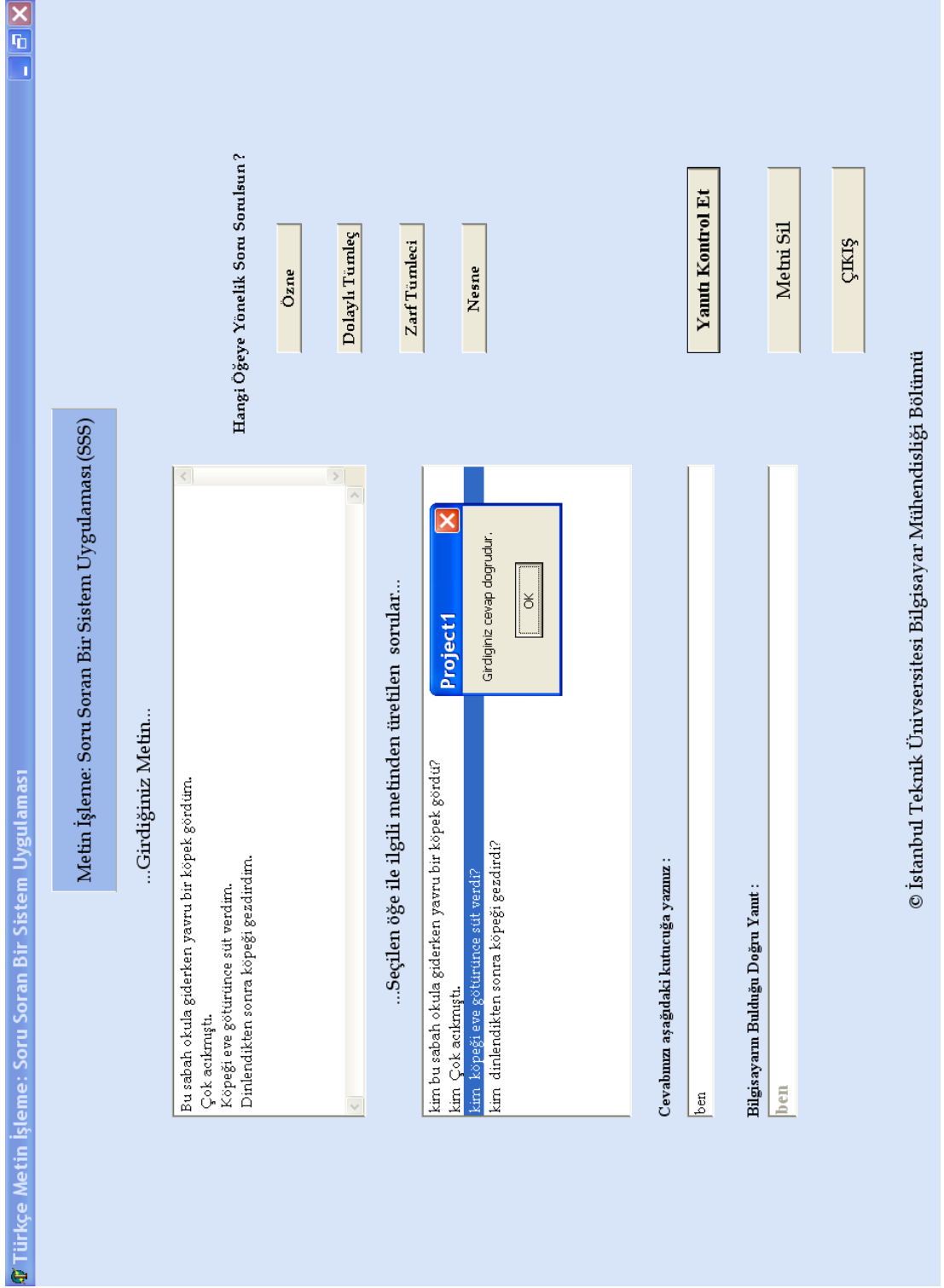
cümlelerin öğelerine ayrılmış hali şöyledir...

**Bu(sf[sf]) sabah(zf[zf])** ----> **zt** / **okul(i[i])** \*a(yon) ----> **dt** / **gid(f[f])** \*e(ys)\*r(gez) \*ken(ken). ----> **yük** /  
**yavru(sf[sf]) bir(sf[sf]) köpek(i[i])** ----> **ö** / **gör(f[f])** \*dü(dgz) \*m(set1). ----> **yük** /  
**Çok(sf[sf])** ----> **zt** / **acık(f[f])** \*mışdı(mgzdgz). ----> **yük** /  
**Köpeğ(i[i])** \*i(bli) ----> **bln** / **ev(i[i])** \*e(yon) ----> **dt** / **götür(f[f])**. ----> **yük** /  
**süt(i[i])** ----> **ö** / **ver(f[f])** \*di(dgz) \*m(set1). ----> **yük** /  
**Dinlen(f[f])**. ----> **yük** /  
**sonra(zf[zf])** ----> **zt** / **köpeğ(i[i])** \*i(bli) ----> **bln** / **gezdir(f[f])** \*di(dgz) \*m(set1). ----> **yük** /

ilk Sayfaya Dön

Şekil 7.8. Girilen Metnin Öğelerine Ayrılmış Şekli

Şekil 7.9'da girilen bir metinden soru çıkarma ve girilen yanıtın doğruluğunu kontrol etme işlemleri gösterilmektedir.



Şekil 7.9. Kullanıcıya soru yöneltilen ve girilen yanıtın kontrol edildiği ekran

## 8. SİSTEM BAŞARISININ ÖLÇÜLMESİ

Türkçe’de soru soran sistem tasarımı konusunda birebir çalışma henüz yapılmadığından, sistemin başarısının ölçülmesinde baz alınacak bir uygulama bulunmamaktadır. Ancak bu konudaki en yakın uygulama “BayBilmiş” adında Yıldız Teknik Üniversitesi’nde geliştirilmiş olan yukarıda da bahsedilmiş olan Internet’ten arama motorlarıyla soru yanıtlayan sistemdir. Bu sistemde, sistem kullanıcılarının günlük diliyle sorduğu sorusunu, önce arama motoru sorgularına dönüştürüp daha sonra Google’a göndermekte ve cevabı sonuç sayfasındaki özet üzerinde ya da sonuç sayfasındaki bağlantılardaki sayfalar üzerinde aramaktadır. Ancak bu uygulamaya bakıldığında soru soran bir sistem değil; soru yanıtlayan bir sistem olduğu ve sadece kelimelerin ek ve köklerine ayrılması safhalarının tez konusuyla benzer olduğu görülmektedir ve bu yüzden bu sistemle karşılaştırma yapmak doğru olmayacaktır. Bu nedenle sistemin başarısı başka bir sistemle karşılaştırma yapılmadan sistemin kendisi baz alınarak, doğru üretilen soruların değerlendirilmesiyle ölçülmüştür. Buna göre bu sistemin başarı testi çok zor ve uzun olduğundan sisteme toplam 13 cümlelik, (sonra sirsistem tarafından 20 basit cümleye dönüştürülmektedir), metinler girilmiş ve Tablo 8.1’de ve 8.2’deki gibi sonuçlar elde edilmiştir.

Tablo 8.1. Sistemin Basit Cümleleri Öğelerine Ayırma Başarısı

Cümlenin Öğeleri	Toplam Öge Sayısı	Doğru Olarak Bulunan Öge Sayısı	Belirtilen Öge İçin Başarı yüzdesi (%)
Özne	15	12	80
Dolaylı Tümleç	11	11	100
Zarf Tümleci	5	3	60
Nesne	9	6	67
Yüklem	24	24	100



Tablo 8.1' göre sistemin basit cümleleri öğelerine ayırma konusundaki ortalama başarıları % 81'dir. Tablo 8.2'de ise şu sonuçlar elde edilmiştir:

Tablo 8.2. Sistemin Soru Üretme Başarısı

Soru Türü	Toplam Üretilen Soru Sayısı	Doğru Üretilen Soru Sayısı	Başarı yüzdesi (%)
Özne Soruları	14	12	85
Dolaylı Tümleç Soruları	10	9	90
Zarf Tümleci Soruları	8	6	75
Nesne Soruları	9	7	77

Tablo 8.2'ye göre sistemin ortalama soru üretme başarıları % 82'dir. Ancak bu sonuçları almak için sisteme önceden verilmiş olan dilbilgisi kurallarına uygun cümlelerin kullanıldığı unutulmamalıdır. Aksi takdirde başarı yüzdesi daha düşük olacaktır.

Tablo 8.1 ve 8.2'de de görüldüğü gibi sistemin soru sorma başarıları öğelerine ayırma başarılarıyla çok yakından ilişkilidir. Öğelerine ayırma başarıları ise kelimelerin ek ve köklerine ayırma başarılarıyla ilişkili olup bunu ölçmek son derece zaman almaktadır. Ancak, ek ve köklerine ayırma başarılarını arttırmak için, 50,000 sözcük kapasiteli bir sözlük kullanmak yerine, küçük ama çok kullanılan sözcüklere sahip bir sözlük kullanmak daha iyi bir sonuç verebilir.

## 9. SONUÇLAR VE TARTIŞMA

Doğal dil işlemenin amacı; analiz eden, anlayan ve doğal insan dillerini oluşturan bilgisayar sistemlerini tasarlamak ve meydana getirmektir. Bu tez çalışmasında doğal dil işleme sistemi Türkçe'ye uygulanarak, çalışmanın bu alanda yapılan çalışmaların geliştirilmesine katkıda bulunacağına inanılmaktadır.

Doğal dil işleme yapay zeka (bilgi gösterimi, planlama, akıl yürütme vb.), biçimsel diller kuramı (dil çözümleme), kuramsal dilbilim ve bilgisayar destekli dilbilim, bilişsel psikoloji gibi çok değişik alanlarda geliştirilmiş, kuram, yöntem ve teknolojileri bir araya getirir.

Metin işleme, anlama, sorgulama ve soru soran ve yanıtlayan sistemlerin temel geliştirilme nedeni kullanıcı dostu sistemler geliştirmek ya da var olan sistemleri kullanıcıya daha yakın kılarak, kullanıcının adeta bir insanla konuşur gibi karşısındaki bilgisayarla iletişimini sağlamaktır.

Piskoloji alanında bilgi tabanına sahip, doğal dil işleme ve metin anlama tabanlı diyalog sistemleri ile gelecekte kişilerin bilgisayarla konuşup, rahatlaması sağlanabilir. Bu tür sistemlere uzman sistemlerin de eklenmesiyle, bu konuşma sonucunda hasta ya da kullanıcı için çeşitli teşhisler ya da kendisine yardımcı olacak çeşitli yorumlar bilgisayar tarafından üretilebilir.

Henüz metin işleme, anlama, sorgulama ve soru soran ve yanıtlayan sistemlerde geline nokta yeterli değildir. Şu anki sistemler genelde tek bir döküman için analizler gerçekleştirmektedir ve kullanılan dökümanlar hep ufak boyuttadır. Gelecekte bir çok dökümanı işleyebilen, çoklu bilgi tabanı sistemlerine ve büyük dökümanları işleme yeteneğine sahip ya da internet üzerinden bu bilgi tabanlarına bağlantılar sağlayıp, mevcut dökümanları tarayabilen sistemler geliştirilmesi planlanmaktadır. Bu da yeni nesil arama motorları geliştirilmesine olanak sağlayacaktır.

Şu an geliştirilmiş sistemlerin hiçbiri %100 doğru çalışmamaktadır. Doğruluk oranı Türkçe gibi sondan eklemeli ve İngilizce'de daha zor olan dillerde daha da

düşmektedir. Anlamsal analiz çalışmaları daha da geliştirilerek, bilgisayarın metinde bire bir geçmeyen bilgiler hakkında yorum yapıp, kullanıcıya cevap vermesi sağlanmaya çalışılmaktadır. Şu anki sistemlerin hiç biri mantıksal olarak anlama, düşünme ve yorum yapma yeteneğine sahip değildir. Ancak bu alanda bir kaç adım atılmış olan projeler de bulunmaktadır.

Bu çalışma daha önce de belirtildiği gibi, ilköğretim düzeyindeki öğrencilerin, okuduklarını anlama düzeylerinin geliştirilmesi amacıyla, kendilerini bilgisayara karşı sınamalarına yardımcı olmak amacıyla geliştirilmiştir.

Çalışmada, girilen metin öncelikle eklerine ayrılmıştır. Daha sonra yan cümleler içeren karmaşık cümleler, basit cümleciklere ayrılarak aralarındaki bağlantılar tespit edilmiştir. Son olarak basit cümleler öğelerine ayrılarak prolog önermelerine çevrilmiş ve prolog veritabanına eklenmiştir.

Kullanıcının bilgisayarın ürettiği sorulara verdiği yanıtlar benzer şekilde öğelerine ayrılarak, doğruluğu kontrol edilmiştir. Kullanıcının yanlış yanıt vermesi durumunda, doğru yanıt kullanıcıya sunulmuştur.

Uygulama projesinde çeşitli konularda eksiklikler mevcuttur. Gerçek anlamda mantıksal düşünüş söz konusu değildir. Örneğin sistem futbolda sert bir hareket sonucunda kırmızı kartın gösterilmesi gerektiğini metinde verilmedikçe düşünemez. Ayrıca sistemin çalışması sırasında “belirtisiz nesne” ve “gizli özne” zaman zaman karışabilmektedir. Bunun nedeni her iki sözcüğün de ek almamasıdır. Bu nedenle sözcüğün eklerine ayrılmış halinden öge ayrımı yapılırken, sistem zaman zaman doğru yanıt karıştırabilmektedir. Bir diğer sorun da bir sözcüğün aynı anda birden fazla türü içermesinde ortaya çıkmaktadır. Örneğin bir sözcük hem isim, hem sıfat, hem de zamir olarak kullanılıyorsa, kelimenin öncesi ve sonrasına ve cümle içindeki konumuna bakılsa da Türkçe’nin esnekliğinden dolayı bazı zamanlarda yanlış sorular üretilmektedir.

Ama eksiklikler doğal dil işleme çalışmaları için son derece doğal bir durumdur. Çünkü doğal dil işleme çalışmaları, özellikle de metin işleme ve anlama tabanlı olanları, son derece zaman alan ve uzun yıllar sonucu geliştirilen çalışmalardır. Örneğin LILOG projesi 60 kişilik bir grubun yıllar süren çalışmaları sonucunda ortaya çıkmıştır.

Geliştirilen bu sistem gelecekte diğer doğal dil işleme çalışmalarıyla birleştirilirse çok daha faydalı uygulamalar yapılabilir. Örneğin bu projeye text-to-speech ve speech-to-text teknikleri eklenirse, bilgisayar girilen metne dair soruları kullanıcıya sesli olarak sorabilir, kullanıcı da bu soruları sesli olarak yanıtlar, yanıt yanlışa bilgisayar kullanıcıya sesli olarak doğru yanıtı iletir ve böylece bilgisayar ve kullanıcı arasında tıpkı insanlarınkine benzer gerçek bir diyalog kurulmuş olur. Bu şekilde geleceğin konuşan, düşünen, anlayan süper bilgisayarlarının geliştirilmesi için ilk adımlar atılmış olur.

Bu konudaki tüm çalışmalar özellikle akademik çevrelerde çok büyük ilgi görmekte ve görmeye de devam edecektir. Çünkü akademisyenler, araştırmacılar ve bilim adamları zamanlarının büyük bir kısmını istedikleri bilgiye ulaşmaya çalışmakla harcamaktadır. Soru sorma ve soru yanıtlama gibi sistemler sayesinde istenen bilgiye ulaşmak için harcanan zaman oldukça kısılacak, bilgiye ulaşım kolaylaşacak ve kişiler, ilgi alanlarına giren konularla daha çok ilgilenebilecek zamanı bulabileceklerdir.

## KAYNAKLAR

- [1] **Matlus Company**, 2003. The Delphi apostle. New York.  
URL: <http://www.matlus.com/scripts/website.dll>
- [2] **Amzi! Inc.**, 2003. Amzi! Prolog+Logic Server. University of Derby Press, United States Of America.  
URL: <http://www.amzi.com>
- [3] **Osman, S.**, 2003. Türk dili ve edebiyatı, Bilge Sanat ve Kültür Yayınları, İstanbul  
URL: [http://www.edebiyatalemi.com/dil\\_bilgisi.htm](http://www.edebiyatalemi.com/dil_bilgisi.htm)
- [4] **Cantu, M.**, 2000. Delphi 5 uygulama geliştirme klavuzu, ALFA Yayınları, İstanbul.
- [5] **Yağimli, M. ve Akar, F.**, 2000. Delphi 5 görsel program tasarımı, BETA Yayınları, İstanbul.
- [6] **Balta, A. ve Yaşar, M.**, 2000. Türk dili ve edebiyatı Lise 1-2, Ertem Basımevi, Ankara.
- [7] **Erenler Y.**, 2004. BLG 505 Doğal dil işleme ders notları, İ.T.Ü. Fen Bilimleri Enstitüsü, İstanbul.
- [8] **Balı, A.**, 2001. İnsan bilgisayar ilişkisinde doğal dil kullanımı, *Bitirme Projesi*, Erciyes Üniversitesi Fen Bilimleri Enstitüsü, Erciyes.  
URL: <http://www.geocities.com/akadirbali/dogaldil/index.html>
- [9] **Bosch, P.**, 1991. Portability of natural language systems, Scientific Center of IBM, Stuttgart.
- [10] **Berri, J.**, 2003. A real world implementation of answer extraction, *PhD Thesis*, University of Zurich, Zurich.  
URL: <http://www.ai.mit.edu/people/jimmylin/papers/Alid98.pdf>
- [11] **Andrew, M.**, 2002. Question answering, *PhD Thesis*, University of Sheffield ,Department of Computer Science, United Kingdom.

- [12] **Garigliano, R. and Nettleton, D. J.**, 2001, *PhD Thesis*, University of Durham, United Kingdom.
- [13] **Tomita, M.**, 1986. Efficient Parsing of NL: A Fast Algorithm for Practical Systems, Kluwer Academic Publishers, Boston, NY, USA.
- [14] **Delmonte R.**, 2002. Text Understanding with GETARUNS for Q/A and Summarization, *PhD. Thesis*, Department of Language Sciences, Università Ca' Foscari, San Marco, VENEZIA.
- [15] **Uszkoreit, H.**, 2002. Computational linguistics and language technology for real life applications, German Research Center For Artificial Intelligence, Germany.  
URL: <http://www.answerbus.com/cgi-bin/answerbus/index.html>
- [16] **Demirli N. Ve İnan Y.**, 2003. Zirvedeki beyinler: Delphi 7, Prestige Yayınları, Ankara.

## **ÖZGEÇMİŞ**

21 Mayıs 1979, İzmir doğumlu Zeki Mocan, orta öğrenimini Hakimiyet-i Milliye İlköğretim Okulu' nda, lise öğrenimini ise İzmir Selma Yiğitalp Lisesi' nde tamamlamıştır. 1997 yılında, Selma Yiğitalp Lisesi' nden mezun olduğu sene, Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği bölümünü kazanmış ve 2002 yılında bu bölümden mezun olmuştur. 2003 senesinde İstanbul Teknik Üniversitesi Bilgisayar Mühendisliği Yüksek Lisans Programı' na girmiş olup, 2005 Ocak'ı itibariyle yüksek lisans tezini sunmaktadır.