

# Türkçe Sohbet Robotu

Sinem TOPÇU<sup>1</sup>, Cansu ŞEN<sup>1</sup>, M.Fatih AMASYALI<sup>1</sup>

## 1. Bilgisayar Mühendisliği Bölümü

Yıldız Teknik Üniversitesi

sinemtopcu.88@gmail.com, cansusen\_36@hotmail.com, mfatih@ce.yildiz.edu.tr

### Özet

*Tarihleri, yapay zekanın varlığının testi için önerilmeleriyle başlayan sohbet robotları günümüzde birçok uygulama alanı (uzaktan eğitim, danışmanlık vb.) bulmuşlardır. Bu çalışmada, genel sohbet amaçlı Türkçe konuşan bir sistem geliştirilmiştir. Sistem AIML teknolojisi üzerine bina edilmiştir. Geliştirilen sistem, kullanıcıdan bilgi alabilme, sohbeti yönlendirebilme, kullanıcıya ait bilgileri kendi cümleleri içinde kullanabilme özelliklerine sahiptir.*

*Anahtar Kelimeler: Sohbet robotu, Diyalog Sistemleri, Aiml, Eliza, Jabberwacky, Alice, Yapay Zeka, Doğal Dil İşleme, Şablon Eşleme.*

### 1. Giriş

Sohbet robotlarının tarihi 1950’de Turing’in ünlü makalesiyle başlamıştır [1]. Turing bir sisteme zeki denebilmesi için kendi adıyla anılan Turing Testini önermiştir. Bu teste göre bir insan, sadece klavye ve monitör kullanılarak yaptığı bir sohbede karşısındakinin insan ya da yazılım olduğunu ayırt edemiyorsa yazılımın zeka sahibi olduğu kabul edilir. Bu kabule birçok karşıt görüş bulunmasına rağmen Turing Testi hala güncelliğini korumaktadır. Her yıl bu test için çeşitli yarışmalar düzenlenmektedir. Bunların en ünlüsü 1991’den beri düzenlenen Loebner yarışmasıdır [2].

Sohbet robotları bir yandan yapay zekanın olabilirliği için ölçüt olarak kullanılırken, diğer yandan günümüzde birçok kullanım alanı ve araştırma alanına kavuşmuştur. Bunlara örnek olarak e-ticaret sitelerinde kullanıcıların sorularını cevaplamak, bir dili yeni öğrenenlere yardımcı olmak, uzaktan eğitim, arama motorları, insanların diyaloglar esnasındaki dinamiklerini anlamak için diyalog analizi, kullanıcı – yazılım (robot) ara yüzlerinin daha doğal (insanlar arası etkileşime yakın) hale getirilmesi sayılabilir [3].

Literatürdeki sohbet robotları geniş kullanım alanı bulmaları için genelde İngilizce için tasarlanmışlardır. Türkçe için ise yapılmış çok az çalışma vardır. Bu bildiriye sunulan projenin

amacı literatürdeki bu eksikliğin azaltılması için Türkçe bir sohbet robotu geliştirmektir.

Geliştirilen programda hedef konuşma türü olarak günlük (tanışma, spor, sinema vb. ) konuşmalar seçilmiştir. Program geliştirilirken diğer dillerde yapılan robotlar ve mimarileri incelenmiş, en esnek ve kolaylıkla geliştirilebilen AIML dili seçilmiştir. Programda, uygun bir cevap bulunmadığında konuyu bilinen bir alana çekme, kaçamak cevaplar verme, konuşmacıyla ilgili sorular sorma ve bu bilgileri kullanma gibi stratejiler üzerinde çalışılmıştır.

### 2. Benzeri Çalışmalar

Sayfa düzeni aşağıdaki kurallara uygun olarak hazırlanmalıdır. Hazır bir taslak (Word, bu taslak) kullanmanız veya ayrıntuların kontrolü için örnek bir dosya takip etmeniz bu gereklilikleri yerine getirmeniz açısından ısrarla önerilir.

Bilinen en eski sohbet robotu (chatbot), ELIZA; 1964 – 1966 yılları arasında Joseph Weizenbaum tarafından yazılmıştır [4]. Eliza, Rogerian tekniğini (hastaya, hastanın söylediklerini tekrar etmek) kullanan bir psikologun simülasyonudur. Örneğin; Eliza’ya bir kullanıcı “Başım ağrıyor.” derse Eliza ona “Demek başın ağrıyor” diyecektir. Eliza’nın tasarım mantığı kullanıcıdan alınan girişi analiz edip, çeşitli dönüşüm şablonları kullanarak bir cevap hazırlamaktır.

Sohbet robotları tarihinin en bilinen bir diğer ürünü ise 1995 senesinde Richard Wallace tarafından yazılan ALICE (Artificial Linguistic Internet Computer Entity)’dir [5]. ALICE’in yapısı AIML mantığına dayanmaktadır. ALICE’in sahip olduğu AIML’ler pattern ve template’lerden oluşmaktadır. ALICE’in temel mantığı kullanıcılarıyla yaptığı konuşmaları sürekli analiz edip altyapısında yer alan AIML dosyalarını geliştirmektir. Ancak bu yapıdaki en büyük problem sistemin herkes tarafından kullanılabilir olması dolayısıyla bazı kullanıcıların sistemi yanlış ve gereksiz bilgilerle doldurmasıdır. ALICE’in ilk uygulaması 1998 yılında JAVA ile Program A adı ile kodlanmıştır. 1999 yılında AIML’ler XML formatına dönüştürülmüş ve bu şekilde AIML geliştiricileri

artmıştır. ALICE, Loebner yarışmasını ve 2000, 2001 ve 2004 yıllarında üç kez en becerikli konuşabilen robot ödülünü kazanmıştır. AIML günümüzde sohbet robotları alanında bir standart haline gelmiştir.

Bir diğer ünlü sohbet robotu olan Jabberwacky, İngiliz programcı Rollo Carpenter tarafından geliştirilmiştir [6]. Jabberwacky'nin teknolojisi var olan diğer sohbet robotlarından farklıdır. Belirli bir kuralı yoktur, kullanıcılarla yapılan tüm konuşmalar bilgi tabanında saklanır. Bu şekilde kullanıcı bir cümle girdiğinde, bu cümle bilgi tabanındaki konuşmalarda aranır ve en uygun / yakın cevap kullanıcıya verilir. Bu şekilde çok çeşitli konular hakkında konuşabilir, birçok yabancı dilde basit cümleler kurabilir. Bu mimarisi nedeniyle Jabberwacky'deki en büyük sorun yine kullanıcıların sistemi suiistimalinden kaynaklanmaktadır.

İngilizce için yapılan bu çalışmaların yanında Türkçe için de yapılmış çalışmalar mevcuttur. Özyurt ve Köse [7] geliştirdikleri sohbet robotunda, kullanıcının girdiği cümlelerin öğelerini bulmakta, daha sonra cümlede eksik olan öğeleri kullanıcıya sorarak sohbeti sürdürmektedir. Sorulacak öğeleri belirlerken cümlenin yüklemine geçişli olup olmadığına da bakılmıştır. Türkçe için ticari anlamda yapılan çalışmaların örnekleri <http://www.chatbots.org/tr> adresinden görülebilir. Bu tarz internet uygulamalarda yer alan sohbet robotları görsel öğelerle desteklenmekte, site ziyaretçilerine sitenin amacına uygun konularda danışmanlık yapmaktadırlar.

### 3. AIML dili

Türkçe Diyalog Sistemi tasarlanırken ALICE'in mimarisi örnek alınmıştır. ALICE'in temel yapısı ise AIML dili kullanılarak oluşturulmuştur. AIML dili (Artificial Intelligence Markup Language) konuşma kurallarını belirleyen XML şemalarıdır [5]. ALICE diyaloglarını üretirken, geliştiricileri tarafından hazırlanmış olan onlarca AIML dosyasını kullanmaktadır. AIML dosyalarında temel 4 etiket kullanılmaktadır:

<aiml>: AIML dokümanını başlatan ve sonlandıran etikettir.

<category>: Bilgi tabanında bulunan herhangi bir bilgi birimini belirtir.

<pattern>: Pattern bir kullanıcının chatbot'a ne söyleyebileceğini basit bir kalıp halinde tutar.

<template>: Kullanıcıya verilecek cevabı içerir.

Basit olarak gösterimi aşağıdaki şekildedir.

```
<aiml>
<category>
  <pattern>....</pattern>
  <template>....</template>
</category>
```

</aiml>

AIML'lerde her category bir pattern ve bir template içerir. Bir category bir başka category tarafından "srai" etiketiyle çağırılabilir. Örneğin;

```
<category>
<pattern>MERHABA</pattern>
<template>Sana da Merhaba</template>
</category>
<category>
<pattern>MERHABALAR</pattern>
<template><srai>MERHABA</srai></template>
</category>
```

şeklinde bir AIML'miz olursa, kullanıcı bize "Merhaba" dediği zaman alacağımız cevap "Sana da Merhaba"dr. Kullanıcı "Merhabalar" dediği zaman program AIML'de arama yapmaya başladığında öncelikle "MERHABALAR" pattern'yle eşleşecektir. Normalde vermesi gereken cevap template etiketlerinin arasında bulunan tümcedir ancak burada template tag'inden sonra srai tag'i geldiği için program "MERHABA" pattern'ini arar ve onun template tag'lerinin arasındaki cevabı verir. Bu kullanımın en büyük avantajı sürekli aynı şeylerin tekrarından kurtulmaktır. Gerçek bir sistemde "Merhaba" gibi bir selamlama sözcüğü için birden fazla cevap bulunmaktadır ve her dilde olduğu gibi Türkçe'de de chat ortamında bazı kısaltmalar mevcuttur. Örneğin; "MRB", "MRHB", "MERABAA" vb. şekillerde girişler olduğunda hepsine aynı template'leri yazmak hem vakit hem yer kaybına neden olur, ayrıca programın gelişim sürecinde "MERHABA" template'inde bir değişiklik yapılmak istendiğinde bunu tümüne uygulamak da verimi düşürücü bir etkidir.

AIML'lerde en çok kullanılan bir diğer tag'de "random"dr. Bu sayede selamlama ayrılma gibi birden çok karşılığı olabilecek sözcüklere farklı cevaplar verilerek, robot'un daha insancıl olması sağlanmaktadır.

```
<category>
<pattern>GORUSURUZ</pattern>
<template>
<random>
<li>Görüşürüz</li>
<li>Seninle konuşmak güzeldi</li>
<li>Yine gel</li>
<li>Hoşçakal</li>
<li>Güle Güle</li>
</random>
</template>
</category>
```

Bu örnekte görüldüğü gibi kullanıcının "Görüşürüz" girişine altı farklı cevaptan biri rastgele seçilip verilebilir.

AIML’lerde en çok örneği görülen bir diğer tag <that> dir.

```
<category>
<pattern>EVET</pattern>
<that>FİLM SEVER MISİN?</that>
<template>En sevdiğin film nedir?</template>
</category>
```

<that> taglerinin arasında bulunan cümle robot’un kullanıcıya sorduğu sorudur. Eğer kullanıcı konuşmanın bir yerinde “EVET” derse bu Evet sözcüğünü ne için dediğini anlamamız gerekir. Buradaki örnekte görüldüğü gibi “EVET” kelimesi geldiğinde program AIML’de bu category’i bulacaktır. Ardından <that> tag’ini görürse bu tag içinde bulunan cümleyi konuşmadaki bir önceki cümleyle karşılaştırır uyuyorsa aynı category’deki template’i cevap olarak verir.

AIML’de konuşmanın daha gerçekçi olması amacıyla kullanılan iki önemli tag daha vardır. Bunlar <get> ve <set>dir. Set tag’inin kullanımına bir örnek;

```
<category>
<pattern>NABER</pattern>
<template>İyilik senden ne haber <set
name="name"/></template>
</category>
```

Program “NABER” pattern’iyle eşleştiğinde cevabı kullanıcıya verebilmek için template kısmına gidecektir. Burada template içerisinde <set> tag’ini bulduğunda ilgili kısımda daha önce kullanıcıdan isim, yaş vb. hangi bilgileri almışsa o özelliğe ait bir tanımla karşılaşacaktır. Burada “name” kullanıldığı için program daha önceden “name” değişkeni ile eşleştirdiği veriyi bu alana ekleyip kullanıcıya cevabı verecektir. <get> tag’inin kullanımı da <set>’e benzer şekildedir.

```
<category>
<pattern>BUGÜN AYIN KACI?</pattern>
<template>Bugün<get date="date"/></template>
</category>
```

Burada program get tag’iyle sistem’den günün tarihini alıp cevabı template’de ilgili kısma ekleyip verecektir.

AIML’de ayrıca, pattern’lerin daha kolay bulunması ve veri fazlalığını azaltmak için çeşitli pattern yazım teknikleri vardır.

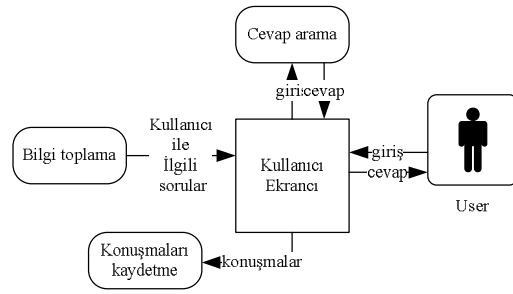
Pattern’lerde en çok kullanılan teknik “\*” koymadır. Bir cümle farklı birkaç formda söylenebiliyorsa bunun yerine \* koyularak arama işlemi kolaylaştırılır. Örneğin “Oyna” kökünü ele alırsak bunun “Oynar, Oynadı, Oynayacak, Oynuyor, vb. ” birden fazla söyleniş biçimi bulunur.

```
<category>
<pattern>FUTBOL OYN*</pattern>
<template>Futbol oynayabilmem için 2
ayak gerek bana.</template>
</category>
```

Bu şekilde bir AIML sayesinde bu gibi köklerin benzer versiyonlarına tek cevaplara karşılık verilebilir.

#### 4. Tasarlanan Sistem

Türkçe Diyalog Sistemi, kayıtlı kullanıcıların bilgisayarla sohbet edebildikleri bir uygulamadır. Buradaki amaç kullanıcıların girişine karşılık bilgisayarın gerçekçi yani bir insanla konuşuyormuş gibi cevaplar verebilmesidir. Sistemin mimarisi Şekil 1’den incelenebilir.



Şekil 1. Sistem Mimarisi.

Türkçe Diyalog Sistemi’nin başrolü kayıtlı kullanıcılarıdır. Bu nedenle sisteme giriş bir kullanıcı adı ve şifreyle olur. Kayıtta kullanıcıdan kullanıcı adı, şifresi ve ad, soyad gibi standart bilgiler istenmektedir. Şekil 2’de sohbet ekranı görülmektedir. Bu ekranda konuşmanın son birkaç cümlesi de gösterilmektedir.

```
Obi: Merhaba
Sen: nasılsın obi
Obi: Bugün pek iyi değilim..
Sen: neden
Obi: Onu da sormayınver, ben sadece bir chatbot'um.
Sen: tamam
Obi: Tamam, anlaştık ;)
Sen: başka ne yaparsın
Obi: Daha ne olsun
Sen: çok komiksin
Obi: Sen komiksin ki bu komik.
Bir şeyler söyle:  
```

Şekil 2. Sohbet Ekranı.

Türkçe Diyalog Sistemi belleğinde o an bulunan aiml dosyalarını kullanarak cevap üretir. Sistemde iki önemli AIML türü vardır. XML şeması olarak aralarında bir fark olmasa da sistemde bulunma süreleri aynı değildir. Bunlardan biri her zaman açık olan selamlama dosyasıdır. Diğer tür ise kullanıcının sözlerine göre belleğe getirilen AIML dosyalarıdır. Tablo 1’de sistemde kullanılan AIML dosyalarının

adları ve içerdikleri şablon (pattern) sayıları verilmiştir.

**Tablo 1.** Kullanılan AIML'ler.

AD	İÇERİK	ŞABLON SAYISI
selamlama	Sıklıkla kullanılan ve herhangi bir konu başlığı altına dâhil edilemeyecek cümleler	364
bilgisayar	Bilgisayarlar ve işletim sistemleri hakkındaki konuşmalar	45
sinema	Popüler sinema tarihi, aktör ve aktrisler	63
muzik	Müzik türleri, sanatçılar, gruplar	35
burc	Burçlar ve özellikleri	40
spor	Spor türleri ve tanınmış oyuncular	40

Kullanıcı, girişi yaptığı andan itibaren program aşağıda verilen sözde koda göre çalışmaktadır.

```
kullanıcı girişi al
if (kullanıcı girişi selamlama.aiml'de)
then cevabı selamlama.aiml'den üret
else
{girişe uygun AIML dosyasını topics.aiml
ile ara
  if (girişe uygun AIML dosyası bulundu)
  then cevabı bulunan AIML dosyasından
  üret
  else
  {if (kullanıcı hakkında eksik bilgi
var)
  then hakkında bir şeyler sor
  else
  {if (hakkındaki veriler ile ilgili soru
sorulmamış)
  then hakkında öğrendiği verilerle
ilgili sorular sor
  else konuyu var olan AIML dosyalarına
çekmek için soru sor
}}}
```

Kullanıcı girişleri bir aiml dosyasının pattern'leri içinde aranırken önce orijinal haliyle daha sonra joker karakter ve kelime halleriyle aranmaktadır. Kullanıcı girişlerinin, pattern'leri içinde ilk arandığı Selamlama.aiml günlük konuşmalarda sık kullandığımız “Merhaba, Nasılsın?, Güle güle, İyi Günler, vb.” ve bunların kullanımında olan değişik formları içermektedir. Selamlama.aiml'de bulunamayan girişler için topics.aiml kullanılarak cümleye uygun aiml dosyası bulunmaya çalışılır. Topics.aiml, her AIML dosyasına ait etiketlerin XML formunda hiyerarşik bir şekilde bulunduğu dosyadır. Örneğin bir cümlede maç, futbol, basketbol, vb. kelimeler geçerse spor.aiml'de; film, aktör,

sinema, beyaz perde, vb. için sinema.aiml'de pattern araması yapılır.

Sistemin veri tabanında her kullanıcıya ait meslek, tuttuğu takım, en sevdiği sanatçı gibi alanlar bulunmaktadır. Cevabın aiml'lerden üretilmemesi durumunda sistem, konuşmayı sürdürürebilmek için kullanıcıdan bu alanları doldurmak için bilgi alma yoluna başvurur. “Sen hangi takımı tutuyordun?”, “Nerde yaşıyorsun?” gibi sorular sorularak bu alanlar doldurulur. Bu alanların dolması halinde sistem yine kullanıcıyı etkilemek için daha önceden elde ettiği verileri soru olarak kullanıcıya yönlendirir. Kullanıcı en sevdiği yemeği sisteme pilav olarak belirttiyse, sistem bunu “Pilav çok seviyorsun da yapabiliyor musun?” gibi cümle içinde kullanarak konuşmanın sürekliliğini sağlamaktadır. Bütün bu aramalar ve bilgi toplama yöntemleri haricinde cevabın yine bulunamaması durumunda sistem konuyu var olan AIML dosyalarına çekmek için onlar hakkında soru sormaktadır.

Sistemde tüm kullanıcılarla yapılan diyaloglar her kişiye ait XML dosyalarında saklanmaktadır. Bu dosyaların analizi sayesinde sistem yöneticileri, sistemin iyileştirilmesi (aiml'lerin güncellenmesi, topics.aiml'nin geliştirilmesi vb.) için çalışmalar yapabileceklerdir.

## 5. Deneysel Sonuçlar

Bu bölümde tasarlanan sohbet robotuyla yapılan diyaloglara çeşitli örnekler verilmiştir.

```
USER:hayır
BOT:Neden hayır?
USER:evet
BOT:Az önce hayır şimdi evet, komiksin.
USER:evet
BOT:Evet, evet, oyun mu oynuyoruz yaa.
USER:hayır
BOT:Bir evet bir hayır, pek kararsız gördüm seni.
USER:ben sıkıldım
BOT:Anlıyorum, ben de bazen sürekli konuşmaktan sıkılıyorum.
```

Bu örnekte görüldüğü gibi sistem, devamlı tekrar eden durumları anlayıp kullanıcıyı başka cevaplar vermeye yönlendirmektedir.

```
USER:hmm
BOT:Hmm, senin burcunu merak ediyorum.
USER:kova
BOT:Sınırsız hayal güçleri olsa da Kova'lar gerçekçi, ileri görüşlü ve akılcıdır.
USER:Ooo bu nedir böyle yaa
BOT:En sevdiğin yemek nedir?
USER:kurufasulye
BOT:Kurufasulye yemeği çok kilo aldırır bence
```

Bu örnekte sistem, kullanıcının “hmm” kelimesinden konuşmanın tıkandığını anlayıp konuyu bildiği alan olan burçlara çekmiştir. Daha sonra kullanıcıdan yine “Ooo bu nedir böyle yaa” cümlesini alınca veri tabanındaki patternlerden biriyle eşleştirememiş bu nedenle kullanıcının veri tabanında ayrılan hakkında kısmında daha önce sorulmayan en sevdiği yemek özelliğini kullanmıştır ve aldığı veriyi cümle içinde kullanarak bunu öğrendiğini göstermiştir.

BOT:INSAAT MUHENDISIYIM işi yapıyorum demiştin. Daha önce ne işleri yapıyordun.

USER:öğrenci oldum sadece

BOT:Ben öğrenci değilim ama öğrencilerin dertlerini iyi bilirim

USER:evet çok dertliydik

BOT:Evet hayır mı oynuyoruz..

USER:oyun mu oynamak istiyorsun

BOT:BARCELONA takımını tutuyorsun.Bana renklerini söyler misin?

USER:bordo mavi

BOT:PILAV yemeğini çok seviyorsun.Bu yemekten bana da yapar mısın?

USER:yaparım da yiyemezsin

BOT:TRABZON'luyum demiştin.Odakiler de senin gibi çok konuşuyor mudur.

USER:hayır

BOT:Neden hayır?

USER:konuşmazlar çok

BOT:ISTANBULdan işe giderken çok trafik var herhalde. Bu agresiflik bundan mı?

USER:agresif değilim

BOT:Tamam öyle değilsin.

Örneğin bu konuşma, kullanıcı hakkında tüm bilgiler toplandıktan sonra cevap bulunamaması durumunda oluşmuştur. Kullanıcını tuttuğu takım, sevdiği yemek gibi özellikleri veri tabanına kaydettiği için bunları kullanarak etkileme yöntemine başvurmuştur.

## 6. Sonuç ve Gelecek Çalışmalar

Sohbet robotları, yapay zekanın varlığının testi için önerildikleri 1950 yılından bu yana bilgisayar dünyası için ilgi çekici alanlardan biri olmuştur. Günümüzde ise uzaktan eğitim, e-ticaret sitelerinde danışmanlık, arama motorları gibi birçok uygulama alanı bulmuşlardır.

Bu projede, Türkçe konuşan bir sohbet robotu geliştirilmiştir. Sistem genel olarak, sohbet robotu çalışmalarında artık bir standart olarak kabul edilen AIML dosyaları ile çalışmaktadır. Kullanıcının söylediği cümlenin tamamı ya da bir bölümünün algılanarak, bu söz için önceden belirlenmiş olan karşılığın cevap olarak atanması, AIML'nin temel mantığıdır.

Sistemle konuşmak, üyelik gerektirmektedir. Bu sayede konuşulan kişiye yönelik hitaplar oluşturulabilmekte ve kişisel bilgiler saklanabilmekte, sonraki konuşmalarda kullanılabilir. Sistem tasarlanırken, dilin Türkçe olması bir takım sorunlar yaratmıştır. Türkçe'nin sondan eklemeli bir dil olması ve özellikle fiillerin çok fazla sayıda çekimi olması, ihtimalleri üstel olarak artırdığından cevap üretme süresini arttırmaktadır. Bu probleme çözüm için kullanıcı girişlerinin morfolojik analizden geçirilmesi düşünülmektedir.

Sistemin temel taşı olan AIML'ler geliştirildikçe sisteminde başarısı artacaktır. Ancak AIML'lerin oluşturulması uzun bir süreç ve emek gerektirmektedir. Bunun için gelecek çalışma olarak AIML'lerin, insan diyaloglarını içeren büyük veri tabanlarından (dizi film altyazıları vb.) otomatik olarak üretilmesi planlanmaktadır.

Sistemin temel taşı olan AIML'ler geliştirildikçe sisteminde başarısı artacaktır. Ancak AIML'lerin oluşturulması uzun bir süreç ve emek gerektirmektedir. Bunun için gelecek çalışma olarak AIML'lerin, insan diyaloglarını içeren büyük veri tabanlarından (dizi film altyazıları vb.) otomatik olarak üretilmesi planlanmaktadır.

## 7. Kaynaklar

- [1] A. Turing, “Computing Machines and Intelligence”, Mind, Vol. LIX, 1950.
- [2] <http://www.loebner.net/Prizef/loebner-prize.html>
- [3] G. Tatai, A Csord'as, 'A. Kiss, L. Laufer and A. Szal'o, “The Chatbot Who Loved Me”, Proceedings of the “Embodied Conversational Characters as Individuals” workshop of the 2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS03), Australia, 2003.
- [4] J. Weizenbaum. ELIZA - A Computer Program for the Study of Natural Language Communication between Man and Machine. Communications of the ACM, 9(1):36-45, 1966.
- [5] R. Wallace, “The Anatomy of A.L.I.C.E”, Proceedings of International LISP Conference, Springer, 2002.
- [6] <http://www.jabberwacky.com>, R. Carpenter, 10 Mart 2010.
- [7] Ö. ÖZYURT, C. KÖSE, “Türkçe Tabanlı Diyalog Sistemi Tasarımı ve Kodlanması”, Elektrik-Elektronik-Bilgisayar Mühendisliği Sempozyumu ve Fuarı-ELECO'2006, pp. 364-368, 2006, Türkiye