Tallinna Ülikool
Digitehnoloogiate Instituut

INTELLIGENTSE ÕPPEKORRALDUSE ABISÜSTEEMI PROTOTÜÜBI ARENDAMINE

Bakalaureusetöö

Autor: Elar Huik

Juhendaja: PhD Erika Matsak

Autor: ..... „ .....„ 2016

Juhendaja: ..... „ .....„ 2016

Instituudi direktor: ..... „ .....„ 2016

Tallinn 2016

SISUKORD

Autorideklaratsioon

Deklareerin, et käesolev bakalaureusetöö on minu töö tulemus ja seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

....

(kuupäev) (autor)

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, Elar Huik (sünnikuupäev: 7. detsember 1990)

1. Annan Tallinna Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose "Intelligentse õppekorralduse abisüsteemi prototüübi arendamine", mille juhendaja on PhD Erika Matsak, säilitamiseks ja üldsusele kättesaadavaks tegemiseks Tallinna Ülikooli Akadeemilise Raamatukogu repositooriumis.
2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tallinnas, _____

allkiri ja kuupäev

Mõisted ja lühendid

AIML - *Artificial Intelligence Markup Language*. Keel, mille abil on võimalik teadmusbaasi kirjutada.

AJAX - *Asynchronous* JavaScript *And* XML.

A.L.I.C.E. - *Artificial Linguistic Internet Computer Entity*.

API - *Application Programming Interface.* Liides olemasoleva programmi kasutamiseks.

CSS - *Cascading Style Sheets*. Keel, millega määratakse veebilehe elementidele stiilid.

HTML - *Hypertext Markup Language*. Keel, millega märgendatakse veebilehti.

HTTP päring - *Hyper-Text Transfer Protocol*. Päring mille kaudu suhtleb veebilehitseja serveriga.

Intelligentne agent - Arvutiprogramm, mis suhtleb inimesega loomulikus keeles.

IT - Infotehnoloogia.

JavaScript - Veebilehtede skriptimise keel.

JSON - *JavaScript Object Notation*. Süntaks andmete hoiustamiseks ja vahetamiseks.

Kasutajakogemus - Emotsioonid, mida inimene kogeb tarkvara kasutamisel.

Kõnesüntees - Inimkõne kunstlik produtseerimine sisend tekstist.

Kõnetuvastus - Tehnoloogia, mille abil tuvastatakse sõnad ja laused, mis vastavad sisendiks olevale inimkõnele.

Loebneri võistlus - Esimene formaalne Turingi test, mille eesmärgiks on leida masin, mille reaktsioonid oleksid eristamatud inimese omadest.

Serveri logi - Fail, või failide kogumik, kuhu talletatakse serverile tehtud päringud.

Lokaalne arendus - Keskkond, kus toimub tarkvara arendus.

Loomulik keel - Keeled, mida inimesed kasutavad üksteisega suhtlemisel.

MySQL - Andmebaasi haldamise süsteem.

PHP - *Hypertext Preprocessor*. Serveripoolne skriptimise keel.

Teadmusbaas - Kogum reeglitest, mille alusel intelligentne agent teeb otsuseid.

Unicode - Rahvusvaheline standard arvutites kirjasüsteemide kodeerimiseks.

Vabavara - Tasuta levitatav tarkvaralahendus.

Veebipõhine - Rakendus, millele ligipääsemiseks on vaja võrguühendust, kasutades HTTP.

Wildcard - Muutuja AIMLis, millega kirjeldatakse tundmatuid sisendeid.

XAMPP - PHP arenduskeskkond.

XML - *Extensible Markup Language*.

Sissejuhatus

Intelligentne tehnoloogia on muutumas tavapäraseks meie ühiskonnas. Maailmas juhtivad ettevõtted nagu Apple, Microsoft ja Google suunavad ressursse, et arendada personaalseid assistente. Need intelligentsed agendid suhtlevad kasutajaga loomulikus keeles, aitavad vastata küsimustele, kasutades andmebaase või otsingumootoreid, saata e-maile, teha kõnesid ja isegi teevad restoranidesse reservatsioone. (Warren, 2014; iOS Siri 2016) 2015. aasta sügisel liitus konkurentsiga ka Facebook oma personaalse assistendiga M. (Hempel, 2015) Tehisintellekti arendamine ja teenusena pakkumine oma kasutajatele on aktuaalne teema.

Apple'i agent Siri ja Google Now on kasutatavad põhiliselt nutitelefonides ja tahvelarvutites. Microsoft'i loodud agent Cortana on kaasas kõikidel Windows 10 platvormidel ja ühildub ka Xbox mängukonsooliga. (Warren, 2015) Facebook'i saab kasutada igas operatsioonisüsteemis, millel on veebilehitseja või rakendusena iOS, Android ja Windows platvormidel. Mainitud agente pakuvad firmad siiani tasuta teenusena, mis tagab tehnoloogia kerge kättesaadavuse.

Autori jaoks on tehisintellektiga seonduv teema huvitav ja usub, et agendid nagu Siri, Google Now, Cortana ja M on eeskujuks tulevastele IT lahendustele, kus eesmärgiks on muuta kasutamiskogemust inimsuhtlusele lähedasemaks. Uued põlvkonnad kasvavad varasemast east üles aina intelligentsemate lahendustega. Ühelt poolt tähendab see, et järgnevate põlvkondade jaoks on täiesti tavapärane suhelda masinaga nagu inimesega ja rakendustelt hakatakse ootama interaktiivsust läbi loomuliku suhtlemise.

Mainitud agentidel on palju funktsionaalsusi, kuid keskseim nendest funtksionaalsustest on suhtlemine kasutajaga loomulikus keeles.

Töö autor on eelnevalt kirjutanud seminaritöö "Uurimus internetipõhistest interaktiivsetest tehisintellekti rakendustest". Uurimuses tõi autor ülevaate kolmest juturoboti loomise süsteemist ja tutvustas ka juturobotite tehnoloogia ajalugu. Uurimuses analüüsiti ja võrreldi AIML, CleverScript ja PersonalityForge agente. Uurimuses selgitati iga agendi töötamise põhimõtteid ja vastavaid keeli, mille abil agentide teadmusbaas kirjutati.

Oma seminaritöös on autor tutvustanud ka võimalikke kasutusi interaktiivsete agentide jaoks, millest üheks on kasutajate assisteerimine veebilehtedel. (Huik, 2015, lk 5)

Esmakordsele külastajale ei ole iga veebilehe navigeerimine intuitiivne ja suuremates keskkondades navigeerimine võib olla aeganõudev ning segadust tekitav. Olukorras, kus kasutaja teab, mis infot tal on vaja, kuid ei oska seda leida, saaks lahendada juturobotiga, mis vastaks kasutaja päringule. Ajendatult sellest probleemist ja huvist tehisintellekti vastu on autoril tekkinud idee arendada Tallinna Ülikooli tarbeks agendi prototüüp, mis aitaks tudengeid lihtsamate õppekorralduse küsimuste puhul.

Käesoleva töö eesmärgiks on tuua ülevaade tehnoloogiatest ja meetoditest, mille abil intelligentse agendi arendamine toimub ja analüüsida agendi arendamisel kasutatud meetodeid.

Eesmärkide saavutamiseks kirjeldab autor esimeses peatükis spetsifikatsioone, mille alusel agent arendatakse ja tehnoloogiaid, mille abil arendamine toimub. Teises peatükis kirjeldab autor agendile vajalikku teadmusbaasi ning selle arendusprotsessi. Kolmandas peatükis analüüsib autor valminud prototüüpi, tuues välja arendamisel valitud meetodite eelistest ja võimalikest probleemidest ning arutleb vajalike etappide üle, mida oleks vaja, et prototüüpi edasi arendada.

1 Spetsifikatsioonid ja tehnoloogiad

Antud peatükis määrab autor nõuded prototüübi jaoks, fikseerimaks tingimused, mille järgi prototüüpi arendada. Peale nõudeid tutvustab autor tehnoloogiaid, mida on otsustanud kasutada agendi arendamiseks ja põhjendab tehnoloogiate kasutamist.

1.1 Prototüübi nõuded

Sõna prototüüp tuleneb kreekakeelsest sõnast *prōtótypon*, mis tähendab "primitiivne vorm". Tarkvaraarenduses on prototüüp algeline töötav mudel lõpptootest või infosüsteemist.

Tarkvara prototüüpimiseks on esialgu vaja välja selgitada nõuded, mida prototüüp peab täitma. Selleks on vaja teada kasutajate sihtgruppi ja mis otstarbeks tarkvara hakatakse kasutama. Kui nõuded on paigas saab selgitada vajalikud funktsionaalsused ja alustada arendusega. (Stanley, 2016)

Käesolevas töös on prototüübi eesmärgiks, suheldes loomulikus keeles, aidata kasutajat Tallinna Ülikooli õppekorraldusega seotud küsimustega. Õppekorraldus on väga lai mõiste ja koosneb mitmetest alamteemadest nagu õppekava, tunniplaan, ainetesse registreerimine, akadeemiline puhkus ja muud teemad. Prototüübis ei ole mõistlik käsitleda kõiki teemasid korraga. Keskenduda võiks põhiliselt ühele või kahele alamteemale, mille raames agent on interaktiivne. Selline teemade arv annab piisava keerukuse ja interaktsioonide arvu.

Tehnoloogiad agendi teadmusbaasi ja funktsioneerimise jaoks peaksid olema vabavaralised. Vabavara puhul on tegemist kättesaadavama lahendusega. Sellisel juhul ei ole tarkvara kättesaadavus tõkestatud rahalistest ressurssidest ja ka lugejal on võimalus arendatud prototüüpi edasi modifitseerida.

Suhtlemine agendiga peaks toimuma veebipõhises keskkonnas. Autoril on õpingutest ja tööga seonduvalt kõige rohkem kogemusi veebiarendamisega kasutades PHP, JavaScript, HTML ja CSS keeli. Veebipõhine agent oleks kasutajatele lihtsam lahendus, sest vajab ligipääsemiseks ainult internetiühendust ja veebilehitsejat.

Prototüübi nõuded saab kokku võtta järgnevalt:

- Agendi ja kasutaja vaheline suhtlus toimub eesti keeles.
- Agent valdab ühte teemat õppekorraldusest.
- Agendi arendamiseks kasutatakse vabavaralisi tehnoloogiaid.
- Agent on veebipõhine.

1.2 AIML

AIML on XML keele ülesehitusest tuletatud märgistuskeel, mille eesmärgiks on võimaldada kindlate mustrite alusel luua teadmus, mis suudaks stiimuli põhjal anda reaktsiooni. Interaktiivse agendi puhul on stiimuliks kasutaja sisend ja reaktsiooniks mingi vastus teadmusbaasist. (Huik, 2015, lk 9; Bush, N., Wallace, R. 2011)

AIML on üle 20 aasta vana keel ja selle aja jooksul on AIML agendid nagu A.L.I.C.E. ja Mitsuku võitnud mitmeid Loebneri võistlusi. (*Home Page of The Loebner Prize in Artificial Intelligence*, 2016; *Loebner Prize*, 2016) Aastate jooksul on tekkinud suur kommuun inimesi, kes arendavad juturoboteid kasutades AIML keelt. Üheks selliseks kommuuniks on Pandorabots. Töö kirjutamisel on keskkonnas www.pandorabots.com üle 225 000 arendaja, kes on kokku loonud üle 280 000 juturoboti. Loodud agente on kasutatud turunduses, hariduses, meelelahutuses, klienditeeninduses, mobiilsetes rakendustes, mänguasjades ja paljudes muudes valdkondades. (*Explore the Pandorabots playground*, 2016) Pandorabots on hea näide sellest, et väga paljudel on huvi rakendada juturoboteid erisuguste ülesannete täitmiseks.

Oma seminaritöös on autor kirjeldanud AIML süntaksi struktuuri ja märgendite funktsionaalsust, mille alusel agendi teadmus luuakse. (Huik, 2015, lk 10-18) Autor soovitab lugeda seminaritööd, et tutvuda AIML keele ülesehituse ja loogikaga.

Käesolevas töös on autor otsustanud kasutada agendi teadmusbaasi loomiseks AIML keelt, sest on seminaritöö raames õppinud piisavalt keele ülesehitust ja loogikat, et osata kirjutada teadmusbaasi.

Interaktiivse agendi loomiseks ei piisa ainult AIML keeles loodud teadmusbaasist. Et toimuks stiimuli-vastusepõhine tegevus kasutaja ja masina vahel on vaja interpretaatorprogrammi.

1.3 Interpretaator

Interpretaator on tarkvara, mis nii öelda elustab agendi. Interpretaator loeb kasutaja sisendit, töötleb seda, seejärel teatud loogikareegleid järgides leiab teadmusbaasist vaste ja tagastab selle kasutajale. Kasutaja sisendi töötlemine on vajalik ühtlustamaks sisendi formaati sellele, mis on teadmusbaasis kirjas. AIML puhul eemaldatakse sisendist kõik kirjavahemärgid ja erimärgid. (Huik, 2015, lk 11)

Eelnevas peatükis mainitud Pandorabots'i keskkonnas saab AIML interpretaatorit kasutada kuutasu eest läbi API. (Huik, 2015, lk 9) Olemas on mitmeid vabavaralisi alternatiive, mis on erinevates programmeermiskeeltes kirjutatud ja nende arendamisele saab kaasa panustada. Populaarsemad AIML interpretaatorid on välja toodud Alicebot kodulehel. (*AIML Implementations*, kuupäev puuub)

1.3.1 *Program* O

Program O on veebipõhine AIML interpretaator, mis võimaldab agendiga suhelda teksti baasil. *Program* O kirjutatud PHP keeles ning kasutab MySQL andmebaasi teadmusbaasi hoiustamiseks ja päringute töötlemiseks.

Tarkvara autoriteks on Elizabeth Perreau ja Dave Morton, kes on oma töö teinud kättesaadavaks GNU Üldise Avaliku Litsentsi tingimustel. (Perreau & Morton, 2016) Loojad tegelevad aktiivselt edasiste funktsionaalsuste arendamisega ja aitavad tarkvaraga esinevaid probleeme lahendada oma Github'i lehel. (*Program O Issues*, 2016) Esimene versioon tarkvarast tehti avalikuks aastal 2009. (*Program-O Version 1*, 2009)

Interpretaatori kasutamiseks on vaja omada serverit, millel on installeeritud PHP. Autor ei leidnud täpsustusi, milline on vanim PHP versioon, millega *Program* O töötaks ja võttis ühendust programmi autoritega. Elizabeth Perreau sõnul on *Program* O tööks vajalik vähemalt PHP versioon 5.3.

Kasutades programmi XAMPP, või sarnaseid vabavaralisi lahendusi, on võimalik PHP serverit lokaalses arvutis kasutada.

Järgnev loetelu on *Program* O tähtsamatest funktsionaalsustest (Perreau & Morton, 2016):

- Toetab mitme agendi kasutamist.
- Toetab Unicode tähemärke alates versioonist 2.2.
- Agenti saab kasutada veebilehel koos AJAX'iga.
- Agenti saab kasutada läbi *Program* O API.
- Vestlused salvestatakse logidesse.

Unicode tähemärkide toetamine tähendab, et teadmusbaasi saab luua kasutades hiina, vene, türgi, kreeka ja paljude muude keelte tähemärke kaasa arvatud eesti keele täpitähti. See funktsionaalsus pole aga alati olemas olnud. Esialgu oli *Program* O mõeldud töötama inglise-keelses keskkonnas toetades vaid ladina tähestikku. (Perreau & Morton, 2016)

Agendi kasutamiseks on AJAX mugav ning hea lahendus. See võimaldab kasutajal suhelda agendiga veebilehel, ilma lehte uuesti laadimata. JavaScript'i abil toimub andmete edasi-tagasi saatmine interpretaatori ja kasutaja vahel. See vähendab serverile tulevaid päringuid lehekülje laadimiseks ja annab parema kasutajakogemuse.

API võimaldab agenti kasutada laialdasemalt kui lihtsalt samas domeenis oleval veebilehel. Kolmandate osapoolte rakendustest või veebilehtedelt saab agendile ligipääsu saates API'le läbi HTTP päringute vajalikud parameetrid. Interpretaator seejärel töötleb sisendit ja kuvab vastust XML või JSON formaadis.

```
var xhttp = new XMLHttpRequest();
xhttp.onreadystatechange = function() {
  if (xhttp.readyState == 4 && xhttp.status == 200) {
    vastus = JSON.parse(xhttp.responseText);
    console.log(vastus.botsay);
  }
};
xhttp.open("GET", "http://api.program-
o.com/v2/chatbot/?bot_id=6&say=what%20is%20your%20name
&convo_id=koodinaide1&format=json", true);
xhttp.send();
```

Koodi näidis 1. *Program* O API poole pöördumine JavaScript'i abil.

Koodi näidises 1 on päring saadetud GET meetodi kaudu koos parameetritega "bot_id", "say", "convo_id" ning "format". Esimene parameeter määrab, millise agendiga tahetakse suhelda ja teine parameeter "say" sisaldab võimalikku kasutaja sisendit. Parameeter "convo_id" on vajalik vestluse salvestamiseks logidesse. (*Program O API Status*, 2016) Viimane parameeter "format" kirjeldab andmete kuju, mida tagastakase päringu saatjale. Koodi näidis 2 illustreerib

tagastatud andmeid, kui formaadiks on JSON ja Koodi näidis 3 toob näite tagastatud XML formaadist.

```
{
  "convo_id":"koodinaide1",
  "usersay":"WHAT IS YOUR NAME",
  "botsay":"My name is Program-O."
}
```

Koodi näidis 2. API vastus JSON formaadis.

```
<program_o>
  <version>2.5.3</version>
  <status>
    <success>1</success>
  </status>
  <bot_id>6</bot_id>
  <bot_name>Program O</bot_name>
  <user_id>91579</user_id>
  <user_name>my friend</user_name>
  <chat>
    <line>
      <input>WHAT IS YOUR NAME</input>
      <response>My name is Program-O.</response>
    </line>
  </chat>
</program_o>
```

Koodi näidis 3. API vastus XML formaadis

Koodi näidises 1 kasutatud API aadress on *Program* O loojate poolt üles seatud testimise eesmärgiks. Kasutamaks enda serveris üles seatud *Program* O API't, peab kasutama vastavat aadressi kuhu API on serveris installeeritud.

Lisaks eelnevatele funktsionaalsustele salvestatakse kõik vestlused logidesse koos unikaalse vestluse identifikaatoriga. Logidest saab jälgida vestluste kulgemist ja on võimalik analüüsida vestluste terviklikkust.

Autor on otsustanud kasutada interpretaatorina *Program* O, kuna see on kättesaadav vabavarana, tarkvara võimaldab agenti kasutada veebipõhiselt, omab mitmeid funktsionaalsusi, mis toetavad sujuvamat arendusprotsessi ja API lubab kasutada agenti ka kolmandate osapoolte rakenduste poolt.

2 Teadmusbaasi arendamine

Käesolevas peatükis kirjutab autor millisele õppekorralduse alamteemale otsustas teadmusbaasi luua ja põhimõtteid ning meetodeid, mille alusel arendus toimus.

2.1 Akadeemiline kalender

Teadmusbaasi arendamise esimene samm on teada vestluse teemat, mille kohta teadmus luua tuleb. Õppekorralduse üheks osaks on akadeemiline kalender, mis määrab iga õppeaasta tegevused. Autor otsustas teadmusbaasi koostamisel lähtuda informatsioonist, mis leidub akadeemilises kalendris. Kalender sisaldab informatsiooni ainetesse registreerimisest, semestrite sisust ja muud tähtsat, mis on õppetegevuse jaoks vajalik. (Akadeemiline kalender, 2016)

Järgmisena on vaja selgitada võimalik vestluse stiil. See on vajalik, et teada kuidas kirjeldada kasutaja sisendeid. Assistendi puhul on eesmärgiks vastata päringutele seoses kalendris oleva infoga. Seega sisaldavad sisendid suurel määral küsisõnu ja võivad sisaldada ka ajamäärust.

2.2 Teadmusbaasi struktuur

AIML võimaldab teadmusosakesi kirjutada globaalselt ja privaatselt. Globaalsed võtmesõnad ei vasta ühelegi kindlale teemale ja ei kirjutata `<topic>` märgendite sisse. Seega vestluse igal hetkel võib kasutaja öelda midagi, mis vastab globaalse teadmusosakese sisendmallile. (Wallace, 2003, lk 12-13; Marietto et al., lk 13)

Privaatsed teadmusosakesed kirjutatakse `<topic>` märgendite sisse ja need on teemapõhised. Nendele teadmusosakestel pääseb agent ligi vaid juhul, kui vestluse käigus on toimunud teema muutus. Iga vestluse alguses on teema määramata ja sisendeid võrreldakse globaalsete teadmusosakestega. Kui vestluse käigus mõni teadmusosake määrab teema, siis järgmiste

sisendite puhul lisandub otsinguprotsessi vastava teema teadmusosakeste hulk. (Wallace, 2003, lk 12-13; Marietto et al., 2013, lk 13)

```
<aiml>
  <category>
    <pattern> MILLAL ON EKSAMID </pattern>
    <template>Eksamid on sügissemestril ja
    kevadsemestril.</template>
  </category>
  <category>
    <pattern> RÄÄGI SÜGISSEMESTRIST </pattern>
    <template> Mida sa tahad teada
      <set name="topic">sügissem</set>estri kohta?
    </template>
  </category>
  <category> ... </category>
  <category> ... </category>
  <topic name="sügissem">
    <category>
      <pattern> MILLAL ON EKSAMID </pattern>
      <template>Sügisel toimuvad eksamid vahenädalal ja
      vahemikul ... </template>
    </category>
    <category> ... </category>
    <category> ... </category>
  </topic>
</aiml>
```

Koodi näidis 4. Globaalsed ja privaatsed (halli taustaga) teadmusosakesed.

Näidises 4 on privaatsed teadmusosakesed halli taustaga. Näite põhjal seletab autor üksikasjalikult, kuidas vestluses toimib globaalsete ja privaatsete teadmusosakeste kasutamine ning mida teema muutus kaasa toob.

Näite 4 põhjal on võimalik järgnev dialoog:

```
Kasutaja: Millal on eksamid?
Agent: Eksamid on sügissemestril ja kevadsemestril.
Kasutaja: Räägi sügissemestrist!
Agent: Mida sa tahad teada sügissemestri kohta?
Kasutaja: Millal on eksamid?
Agent: Sügisel toimuvad eksamid vahenädalal ja vahemikul ...
```

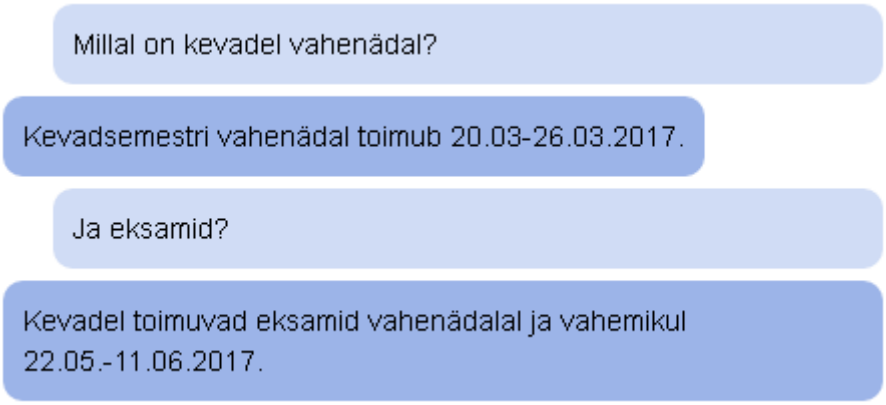
Vestluse alguses ei ole aktiivset teemat määratud. Esimesele sisendile vastab agent globaalsest teadmusosakesest nagu koodi näidises näha on. Järgmise sisendi puhul nõuab kasutaja agendil rääkida sügissemestri kohta. Teadmusosake, mis vastab sellele sisendile sisaldab endas `<set>` märgendeid, millega määratakse teema nimeks "sügissem". Peale seda

küsib kasutaja uuesti "Millal on eksamid?" ja agent vastab talle palju täpsemalt kui esimesel korral.

Teema muutuse peale muutub ka loogika, mille puhul väärtustatakse mingit teadmusosakest. Näidises 4 on näha, et samasugune sisend on defineeritud globaalselt ja "sügissem" teema sees. Kui kasutaja sisendile sobiks globaalne ja mingi teemapõhine vastus, siis agent valib teemale vastava teadmusosakese.

Selline loogika on tähtis, sest aitab simuleerida inimvestluses esinevat kontekstiteadlikkust ja mäletamist. Korraga saab aktiivne olla ainult 1 teema. See omakorda tekitab võimaluse kasutada erinevate teemade sees sarnaseid või ühesuguseid sisendmalle.

Teadmusbaasi kirjutamisel on autor lähtunud ideest kirjutada teadmus võimalikult teemapõhiselt. Niiviisi saab agent teemaga seonduva info kirjeldada teema sisse. Lisaks on võimalik kirjeldada väga üldisi sisendeid, mis koosnevad üksikust tegusõnast, küsisõnast, nimisõnast või paari sõna kombinatsioonist. Teemapõhise struktuuri puhul on globaalsete teadmusosakeste ülesanne aru saada vestluses esinevaid teema muutusi.



Joonis 1. Vestlus inimese (helesinine) ja prototüübi (tumesinine) vahel.

Joonis 1 vestluses saab agent esimesest sisendist aru, et kasutaja tahab teada kevadsemestri vahenädala kohta. Järgmine kasutaja sisend on ebamäärane. Kui kasutaja oleks öelnud "Ja eksamid?" esimese asjana vestluses, ei saaks agent aru, mida kasutaja pärib. Samuti ei saaks ka inimene sellest aru, sest puudub igasugune kontekst. Kuna teemaks on juba kevadsemester, siis agent mõistab, et tahetakse teada, millal on eksamid kevadsemestril. Antud vestlus on ka hea näide konteksti vajalikkusest ja kuidas agent selle abil saab kasutaja sisendist aru.

2.3 Rekursiivne võtmesõnade töötlemine

Käesolevas peatükis seletab autor, kuidas teadmusosakesed töötavad funktsionaalselt, et toetada teemapõhist struktuuri.

AIML sisendmallide kirjutamiseks on kaks meetodit. Ühel juhul võib kirjutada sisendi staatiliselt, ehk sisendmall peab üks ühele vastama kasutaja sisendiga. Teisel juhul saab sisendi sisu kirjeldada üldisemalt ehk dünaamiliselt, kasutades *wildcard*'e. (Marietto et. al,2013, lk 4) Koodi näidis 5 sisaldab näidet mõlemast meetodist.

```
<category>
  <pattern>
   KUIDAS SAAB ÕISIS AINETESSE REGISTREERIDA
  </pattern>
  <template> ... </template>
</category>
<category>
  <pattern> * ÕISIS * </pattern>
  <template> ... </template>
</category>
```

Koodi näidis 5. Staatiline ja dünaamiline sisend.

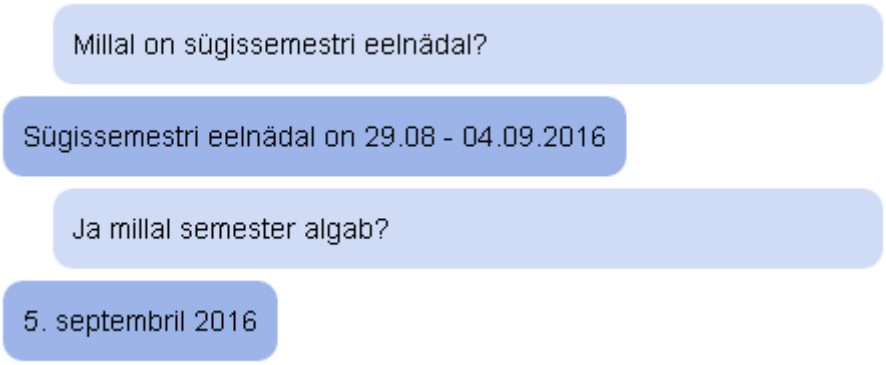
Kui kasutaja küsib "Kuidas saab ÕISis ainetesse registreerida?" siis sobivad, näidisest 5, vasteteks mõlemad teadmusosakesed. Eesti keeles on võimalik sama mõttega lauset küsida erinevatel viisidel. Juhul, kui kasutaja sisend oleks "Kuidas ma saaksin ÕISis ainetesse registreerida?", siis sobiks ainult teadmusosake, mille sisendmall on `<pattern> * ÕISIS * </pattern>`. Tärnid omistavad endale sisendist vastavalt "Kuidas ma saaksin" ja "ainetesse registreerida". Dünaamilise sisendmalli puhul on kirjeldatud suuremal hulgal võimalikke kasutajasisendeid.

Teemapõhise teadmusbaasi loomiseks peab igale teemale leidma kindla sõna või sõnade kombinatsiooni, mille järgi agent tuvastab teema muutuse. See on võtmesõna, millega saame teada vestluse üldise konteksti. Teema siseselt peab defineerima järgmise hulga võtmesõnu, millega saab täpsustada informatsiooni sisu üldisest kontekstist. Koodi näidise 5 puhul on näha, kuidas võtmesõnu sisaldavad sisendid peaksid olema kirjeldatud - dünaamiliselt.

Võtmesõnade kasutamine on vaja, et agent saaks sisendeid järgneva loogika põhjal uurida:

1. Sisendist leitakse teemale vastav võtmesõna.
2. Agent määrab teema muutuse.
3. Ülejäänud sisendist otsitakse võtmesõnu, mis on teemas kirjeldatud.
4. Võtmesõnade abil täpsustatakse mingi arvu sammude haaval informatsiooni sisu.
5. Täpsustamise tulemusena leitakse vastus.
6. Agent tagastab vastuse.

Koodi näidise 6 ja joonises 2 oleva vestluse põhjal selgitab autor, kuidas on teadmusosakesed teadmusbaasis kirjutatud ning kuidas rekursiivselt toimub informatsiooni täpsustamine võtmesõnade abil.



Joonis 2. Lühike vestlus prototüübiga.

```
  <category>
    <pattern>_ SÜGISSEMESTRI _</pattern>
    <template>
      <srai>
        <set name="topic">Sügissemester</set> <star index="1"/>
        <star index="2"/>
      </srai>
    </template>
  </category>
<topic name="Sügissemester">
  <category>
    <pattern>* EELNÄDAL</pattern>
    <template><srai>EELN <star index="1"/></srai></template>
  </category>
  <category>
    <pattern>EELN * MILLAL *</pattern>
    <template><srai>EELN KESTVUS</srai></template>
  </category>
  <category>
    <pattern>EELN KESTVUS</pattern>
    <template>Sügissemestri eelnädal on 29.08 - 04.09.2016</template>
  </category>
</topic>
```

Koodi näidis 6. Joonis 2 vestlust võimaldav teadmusbaasi osa.

Joonisel 2 olevas vestluses on esimene kasutaja sisend "Millal on sügissemestri eelnädal?". Esialgu ei ole teemat määratud ja sisendist tuvastatakse sõna "sügissemestri" sisendmalliga `<pattern>_ SÜGISSEMESTRI _</pattern>`. Märgid "_" on *wildcard*'id.

Kuna teadmusosake kirjeldab teemat, siis esimese asjana määratakse `<set>` märgenditega vastav teema. `<star/>` märgendid võtavad enda rolliks kõik sõnad, mis asuvad sõnast "sügissemestri" vasakul ja paremal. `<srai>` funktsioon tekitab uue sisendi, millele inerpretaator hakkab vastet otsima. Uus sisend näeb välja selline: "Sügissemester Millal on eelnädal".

Teema "Sügissemester" on nüüd aktiivne ja järgmisena leitakse vaste teadmusosakesest sisendmalliga `<pattern>* EELNÄDAL</pattern>`. Samuti sisaldab see teadmusosake järgmist `<srai>` funktsiooni, kus sisend muudetakse kujule "EELN Sügissemester Millal on". Sellele sisendile vastab omakorda sisendmall `<pattern>EELN * MILLAL *</pattern>`. Nüüd toimub viimane `<srai>` päring sisendiga "EELN KESTVUS", millele leitakse vaste, kus on lõplik vastus sisendile.

Agent on kasutaja sisendist jõudnud vastuseni rekursiivselt, tuvastades järk järgult võtmesõnu, et täpsustada sisendis peituv informatsioon.

3 Analüüs

Esimeses alampeatükis teeb autor ülevaate valminud prototüübist. Järgmistes alampeatükkides analüüsib autor arendamiseks kasutatud meetodite eeliseid ja murekohti. Viimases alampeatükis selgitab autor edasised arendusprotsessid, mida oleks vaja, et prototüübist saaks reaalselt kasutatav agent.

3.1 Valminud prototüüp

Agent suudab kasutajatega suhelda seoses akadeemilises kalendris põhineval informatsioonil. Kalendris olev info on teadmusbaasis jaotatud eraldi teemadeks.

Kasutaja sisendile vastuse leidmiseks tegutseb agent rekursiivsel viisil. Esialgsest sisendist algab järk-järguline protsess, mille lõpuks leitakse sisendi tähendusele vastav reaktsioon.

Järgnevalt on loetletud teemad ja nendega seonduv informatsioon, mis on teadmusbaasis kirjeldatud:

- Kevadsemester ja sügissemester.
    - Semestrite algus, kestvus, lõpp.
    - Eelnädalate ajakava ja sisu.
    - Eksamisessioonide ajakava.
    - Toetuste ja stipendiumite taotlemine.
- Vaheajad.
    - Algus, kestvus, lõpp.
- Õppeinfosüsteem.
    - Asukoht.
    - Tõrked.
    - Eksamitele registreerumine.
        - Ajaline info.
        - Juhend.

Lisa 1 sisaldab näidis vestlusi agendiga, kus on näha, milliste sisendite puhul agent vastab.

Lugejal on prototüübiga võimalik suhelda aadressil http://ehco.planet.ee/agent/ ja agendi teadmusbaasi lähtekood on saadaval aadressil https://github.com/Zaehiel/Teadmusbaas.

3.2 Eelised

3.2.1 Teadmusbaasi struktuur

Prototüübi teadmusbaas on kirjutatud teemapõhiselt. Kõik teemaga seonduvad teadmusosakesed on kättesaadavad, kui vastav teema on aktiivne. Kõik teadmusosakesed, mis ei sisalda teemaga seonduvat informatsiooni, aga määravad vestluse käigus teema muutust, pole kirjutatud ühegi teema sisse.

Sellise struktuuri abil on võimalik:

- luua loomulikum vestlus;
- lihtsamalt sisendmalle kirjeldada;
- kergemalt sarnaseid teemasid käsitleda.

Loomulikum vestlus tuleneb agendi võimekusest tuvastada, millise teema kontekst on vestluses asjakohane. Joonis 1 ja 2 on head näited kontekstiteadlikkusest.

Kontekstipõhine teadmus on ka põhjus, miks saab kirjeldada väga üldisi sisendmalle. Arendamise seisukohalt on see hea, sest kirjeldada on vaja vähem sisendmalle. Samuti aitab see muuta vestlust loomulikumaks.

Aktiivsena saab olla ainult üks teema ja ei saa tekkida olukorda, kus vestluse käigus kasutaja sisendile leitakse spontaanselt vaste mitteaktiivse teema seest. Küsides agendilt kevadsemestri algamise kohta ja siis küsides "Millal lõpeb?", vastab agent alati kevadsemestri kontekstist lähtudes, kuigi sügissemestri teemas võib olla kirjeldatud teadmusosakse samasuguse sisendi jaoks.

3.2.2 Dünaamiline sisendmall

Teadmusbaasi koostamisel on üheks suurimaks raskuseks võimalike sisendite määratlemine. Sõnade asetus võib olla väga erinev mingi kindla mõtte edastamiseks. Eesti keeles on ka palju käändeid, mis omakorda suurendab võimalikku sisendite hulka, mida on vaja kirjeldada.

Dünaamilised sisendmallid võimaldavad:

- kirjeldada sisendeid üldisemalt.
- vähendada sisendmallide arvu.
- kiirendada arendusprotsessi.
- vähendada kirjavigade riski.

Koodi näidises 8 on kirjeldatud kahe reaga kõik sisendid, mis on näidises 7.

```
<pattern> MILLAL SAAB STIPENDIUMI TAOTLEDA </pattern>
<pattern> MILLAL SAAB TAOTLEDA STIPENDIUMI </pattern>
<pattern> MILLAL VÕIB STIPENDIUMI TAOTLEDA </pattern>
<pattern> MILLAL VÕIB TAOTLEDA STIPENDIUMI </pattern>
<pattern> MIS KUUPÄEVADEL SAAB STIPENDIUMI TAOTLEDA </pattern>
<pattern> MIS KUUPÄEVADEL SAAB TAOTLEDA STIPENDIUMI </pattern>
<pattern> MIS AJAL SAAB STIPENDIUMI TAOTLEDA </pattern>
<pattern> MIS AJAL SAAB TAOTLEDA STIPENDIUMI </pattern>
<pattern> MILLAL SAAB TAOTLEDA STIPENDIUMI </pattern>
<pattern> MILLAL VÕIB STIPENDIUMI TAOTLEDA </pattern>
<pattern> MILLAL VÕIB TAOTLEDA STIPENDIUMI </pattern>
```

Koodi näidis 7. Sarnased sisendid.

```
<pattern> MILLAL * TAOTLEDA </pattern>
<pattern> MILLAL * TAOTLEDA * </pattern>
```

Koodi näidis 8. Lihtsustatud sisendid.

Sisendmallide hulka saab vähendada kasutades *wildcard*'e, et kirjeldada kõiki võimalikke viise ühe mõtte ütlemiseks. Vaja on kirjeldada ainult kõige tähtsamad sõnad, nii öelda võtmesõnad, millest saab järeldada võimaliku lause informatsiooni.

Sellisel meetodil on teadmusbaasi arendus kiirem. Sisendmallides on vaja kirjeldada võtmesõnade asukohad ja võimalikud käändevormid.

Dünaamilised sisendid lahendavad ka olukorra, kus kasutaja peaks agendi poole pöörudma väga vähese informatsiooniga. Joonisel 1 saab kasutaja sisendist "Ja eksamid?" agent aru tänu sisendmallile `<pattern>* EKSAMID</pattern>`.

Dünaamiliste sisendite puhul väheneb ka oht, et kirjavigade tõttu ei leita teadmusbaasist vastet. Nii kaua kuni kasutaja on võtmesõnad korrektselt kirjutanud, tuvastatakse vastav sisendmall ja agent vastab adekvaatse infoga.

3.2.3 Rekursiivne otsing

`<srai>` funktsiooni abil saab muuta sisendist info otsimise järk-järguliseks protsessiks, kus otsitakse järgnevaid seoseid, et kokku saada täielik informatsioon. Tähtis osa sellest protsessist on võimalus edastatavat sisendit muuta. Selle abil saab täpustada informatsiooni hargnemist.

Koodi näidises 8 on kirjeldatud taotlemisega seotud informatsioon. Ülikoolis saab taotleda stipendiumi ja toetusi. Kuna kirjeldatud sisendmall on väga üldine, siis on vaja täpsustada, millise taotluse kohta päritakse. `<srai>` abil otsitakse siis järgnevalt sisendist, kas päring on stipendiumide või toetuste kohta. Kui on jõutud lõpliku järelduseni toimub viimane `<srai>` kasutus, mis viib lõpliku vastuseni.

Selline järjestikune täpsustamine võtmesõnade abil, vähendab vajalikku sisendmallide hulka.

3.3 Probleemid

3.3.1 Teema muutus konteksti põhjal

Kõik võtmesõnad, mille abil agent tuvastab, millisest teemast räägitakse, on globaalselt kirjas. Iga sisendi puhul, olenemata aktiivse teema olemasolust, kontrollitakse, kas kasutaja on agendi poole pöördunud kasutades teemale vastavat võtmesõna. Kui selline võtmesõna on olemas, muudab agent aktiivset teemat.

Sarnaste teemade, nagu kevad- ja sügissemester puhul võib tekkida vestlus, kus kasutaja küsib sügisel toimuvate eksamite kohta. Agent tuvastab, et räägitakse sügissemestrist ja ütleb kasutajale, millal sügissemestri eksamid on. Kui järgmise sisendi puhul kasutaja peaks küsima "Aga kevadel?" siis agent küll tuvastab, et teemaks on nüüd kevadsemester, kuid ülejäänud sisend ei sisalda ühtegi võtmesõna, millega saaks kirjeldada kogu informatsiooni sisu. Inimesel oleks lihtne aru saada, et nüüd tahetakse kevadel toimuvate eksamite kohta teada, aga agent seda seost ei tee.

Probleemi lahendamiseks on vaja tuvastada situatsioonid, kus ühes teemas võib tekkida üleminek teise teemasse. Seejuures teema muutusest arusaamine peab toimuma eelneva konteksti põjal. Siis peab kirjeldama teema sees võimalikud sisendmallid, mis tekitavad teema muutust. Sellised sisendmallid tuleb staatiliselt kirjeldada, sest ainult sel juhul on tagatud täielik vastavus kasutaja sisendiga. Kui kasutaja sisend vastab täielikult mingile sisendmallile, siis tagastatakse ainult selle teadmusosakese sees olev vastus.

```
<category>
    <pattern>_ KEVADEL</pattern>
    <template>
       <srai><set name="topic">Kevadsemester</set> <star/></srai>
    </template>
  </category>
<topic name="Sügissemester">
  <category>
    <pattern>AGA KEVADEL</pattern>
    <template>
      <that>Eksamid on sügisel vahenädalal ja vahemikul ...</that>
      <srai><set name="topic">Kevadsemester</set> ALGAB EKSAMID</srai>
    </template>
  </category>
</topic>
```

Koodi näidis 9. Teema vahetus teema sees.

Koodi näidis 9 on võimalik lahendus eelnevalt kirjeldatud olukorrale. Kasutatud on ka `<that>` funktsiooni, mis kontrollib, et agent on eelnevalt öelnud vastava sisendi. Selle abil saab kindlustada kontekstist arusaamise ja vastavalt määrata `<srai>` sisu.

Probleemile on lahendus, aga see suurendab teadmusbaasi koostamise keerukust ja ka sisendmallide mahtu. Sellised situatsioonid ei ole ka alati läbinähtavad ja seetõttu on raske võimalikke seoseid ette näha.

3.3.2 Kirjavead

Kirjeldades teadmusbaasis ainult vajalikke võtmesõnu, on vähem tõenäolisem, et agent ei tuvasta sisendi tähendust, kui mõni kõrvaline sõna on valesti kirjutatud.

Siiski on võimalus, et inimliku eksimise tõttu võib klaviatuuril vajutada valele nupule ja sisendis olev sõna on agendi jaoks tundmatu. Nutiseadmete puhul on isegi kergem vajutada valele klahvile, sest puutetundlikul ekraanil võivad klahvid olla väga lähestikku.

Piisab, et kasutaja sisendis oleks üks võtmesõnadest valesti kirjutatud ja agent ei suuda järeldada, mis on lause tähendus.

3.3.3 Keerulised sisendid

Agent suudab aru saada lihtsamatest lausetest, kus konkreetselt küsitakse ühe asja kohta. Näiteks küsimuse "Millal on eelnädal?" puhul on õige, et agent teavitab vastu eelnädala toimumise kuupäevadest. Kui kasutaja pöördub agendi poole liitlausega, milles küsitakse kahe või enama asja kohta, vastab agent ainult ühele nendest asjadest. Näiteks küsimuse "Millal on eelnädal ja mis üritused on eelnädalal?" puhul vastab agent samamoodi nagu eelmise küsimuse puhul.

Probleemiks on võtmesõnade tuvastamise järjekord. Liitlauses on võtmesõnadeks "eelnädal", "millal" ja "üritused". Esialgu tuvastatakse, et informatsiooni sisu on seotud mingi semestri eelnädalaga. Peale sisendi muutmist `<srai>` päringu jaoks, esineb "millal" enne "üritused" ja minnakse mööda jada, mis viib eelnädalate ajalise toimumise infoni. Kui sisend oleks kujul "Mis üritused on eelnädalal ja millal see on?" siis agent teavitab ainult toimuvatest tegevustest.

3.4 Analüüsi kokkuvõte

Valminud prototüüp oskab lihtsamate sisendite puhul suhelda kasutajaga õppeaasta tegevustest. Tegevused on jaotatud struktuuri poolest 4 teemasse, mille sees on kirjeldatud teema sisuline informatsioon.

Teemapõhine struktuur tagab agendi kontekstiteadlikkuse, mis on üks olulisi oskusi suhtluses. Samuti lihtsustab sellise struktuuri kasutamine sisendmallide kirjutamist ja sarnaste teemade käsitlemist, sest ei pea muretsema, et ühele teemale omane informatsioon oleks kättesaadav teises teemas.

Teadmuse töötamiseks on teadmusosakesed kirjutatud dünaamiliselt, mille sisendmalli sisuks on võtmesõna. Võtmesõnadega toimub sisendist arusaamine rekursiivse protsessina. See tähendab, et ükski kasutaja terviklik sisend ei ole välja kirjutatud. Kasutaja sisendi tähendusest saadakse aru peale järk-järgulist analüüsimist. Antud meetod lühendab sisendmallide sisu ja vähendab vajalikke teadmusosakeste arvu.

Üheks mureks agendi kasutamisel on võimalik kirjavigade esinemine kasutaja sisendis. Dünaamiliste sisenendmallidega on küll võimalik vähendada seda riski, kuid mitte täielikult, sest võtmesõnad peab ikkagi kirjutama staatiliselt.

Sarnaste teemade käsitlemisel lisandub teadmusbaasi koostamisele keerukus teemadevaheliste interaktsioonide kirjeldamiseks. Probleemile leidub lahendus, mis vajab teadmusosakeste kirjeldamist staatiliselt ja kasutades eelnevalt öeldud lause sisu, kui konteksti, mille alusel tuvastatakse teemavahetus.

Keerulisemate lausete puhul, kus kasutaja küsib mitme asja kohta korraga, ei suuda agent tuvastada kõiki vastuseid. See tuleneb võtmesõnade sisselugemise järjekorrast.

3.5 Edasine arendus

3.5.1 Teadmusbaasi täiendamine ja testimine

Esimene samm terviklikuma agendi jaoks on olemasoleva teadmuse täiendamine ja uue teadmuse lisamine.

Iga uue teema kohta peab:

1. Selgitama teema jutuainete sisu.
2. Selgitama võimaliku vestluse stiili.
3. Defineerima võtmesõnad iga jutuaine jaoks.
4. Kirjeldama rekursiivse meetodiga vestluse sisendid.
5. Testima vestluse toimimist.
6. Täiustama olemasolevaid või kirjeldama uusi sisendmalle kui testimine tuvastas puudujääke.

Piisava hulga teemade olemasolul tuleks testimisse kaasata tudengid. Logidest saab ülevaate kõikidest testimisel toimunud vestlustest ja tuvastada puuduvad sisendmallid.

Testijatelt võiks küsida ettepanekuid või ootusi agendi kohta, et selgitada millised olemasolevad teemad on nende jaoks tähtsamad. Lisaks võiks ka uurida, kas on teemasid, mis ei ole seotud õppekorraldusega, aga on vajalikud tundengitele.

3.5.2 Kõnetuvastus ja kõnesüntees

Eesti Keeletehnoloogia (EKT) projektide raames arendatakse tarkvara kõnetuvastuse ja kõnesünteesi otstarbeks. (Eesti Keeletehnoloogia Riiklik Programm, 2016) Olemasoleva tarkvara põhjal on loodud mitmeid töötavaid rakendusi nagu "Arvutaja" ja "Uudistelugeja".

Kõnesünteesi ja kõnetuvastuse tarkvara ning lähtekood on EKT teinud avalikult kättesaadavaks. (Eesti Keeletehnoloogia Riiklik Programm, 2016; Eesti Keele Instituut, 2016)

EKT poolt loodud tarkvara implementeerine tähendaks inimlikumat agenti, mis suudaks suhelda ka vaegnägijatega.

Kokkuvõte

Bakalaureusetöö eesmärgiks oli tuua ülevaade tehnoloogiatest ja meetoditest, mille abil on võimalik arendada intelligentset agenti ning analüüsida kasutatud meetodeid. Eesmärkide saavutamiseks arendas autor intelligentse agendi prototüübi.

Prototüübile vajalik teadmusbaas arendati AIML keeles. Teadmusbaasi töötamiseks kasutati interpretaatorit *Program* O ja tutvustati tarkvara funktsionaalsusi. Arendusprotsessis kirjeldas autor teadmusbaasi koostamise struktuuri, sisendmallide kirjutamise põhimõtteid ja töötamise loogikat. Analüüsis tõi välja kasutatud meetodite eelised ja probleemid. Tööst selgus, et kasutatud meetodite eelisteks on tugevam kontekstiteadlikkus ja loomulikum suhtlus ning meetodid lihtsustavad teadmusbaasi arendamist. Probleemaatiline on raskemate sisendite töötlemine ja kirjavigade tõttu ei ole võimalik sisenditest informatsiooni tuvastada.

Käesoleva töö raames valmis intelligentse agendi prototüüp, mis suhtleb loomulikus keeles ja vastab päringutele Tallinna Ülikooli akadeemilises kalendris oleva infoga. Agendiga saab suhelda tekstipõhiselt kasutades veebilehitsejat. Protoüüpi on võimalik edasi arendada ja autor on ka teinud ülevaate etappidest, mille alusel uue teadmuse loomine peaks toimuma.

Summary

Developing a Prototype of Intelligent Assistant for Study Regulations

The objective of this thesis was to give insight about the technologies and methods, by which it is possible to develop an intelligent agent, and to analyze the methods, that were used for development. To achieve these goals, the author developed a prototype assistant.

The thesis consists of three parts. In the first section, the author defines the requirements that the prototype must fulfill and introduces the technologies that will be used. AIML was used to write the knowledge base and Program O was used as the interpreter for the knowledge base. The second part consists of the development process, in which the author describes how the units of knowledge were structured. Furthermore, the author describes, in detail, the logic that was used to write the patterns and templates, so that the agent would react naturally and within the context of the conversation. Last but not least, the author analyzed the methods that were used in the development process. The methods used, allow for the creation of an agent with strong contextual awareness while simplifying the development process. However, the agent will not be able to understand sophisticated inputs from the users and will be unable to understand any inputs that contain grammatically incorrect keywords.

As part of this thesis, a prototype assistant was developed. The assistant is a web-based application and interacts with any users via text inputs. The source code for the assistant can be found here: https://github.com/Zaehiel/Teadmusbaas. The prototype helps students about questions regarding the events and dates in the academical calendar. For further development, the author has also outlined the necessary steps that need to be follow when developing the knowledge base.

Kirjanduse loetelu

Warren, T. (2014, 4. aprill). *The story of Cortana, Microsoft's Siri killer*. Loetud 2. aprill 2016 aadressil http://www.theverge.com/2014/4/2/5570866/cortana-windows-phone-8-1-digital-assistant

iOS Siri. (2016) Loetud 10. aprill 2016 aadrssil http://www.apple.com/ios/siri/ - Siri product page koos funktsionaalsuste ülevaadetega

Hempel, J. (2015, 26. aguust). *Facebook Launches M, Its Bold Answer to Siri and Cortana.* Loetud 9. aprill 2016 aadressil http://www.wired.com/2015/08/facebook-launches-m-new-kind-virtual-assistant/

Warren, T. (2015, 15. juuni). *Xbox One dashboard update includes a huge new design and Cortana*. Loetud 2. aprill 2016 aadressil http://www.theverge.com/2015/6/15/8786501/microsoft-xbox-one-dashboard-update-features

Huik, E. (2015). *Uurimus internetipõhistest interaktiivsetest tehisinetellekti rakendustest* (seminaritöö). Loetud aadressil http://www.cs.tlu.ee/teemaderegister/

Bush, N., Wallace, R. (2011). *Artificial Intelligence Markup Language (AIML) Version 1.0.1*. Loetud 12. aprill 2016 aadressil http://www.alicebot.org/TR/2001/WD-aiml/

Explore the Pandorabots playground. (2016). Loetud 5. aprill 2016 aadressil https://playground.pandorabots.com/en/

Home Page of The Loebner Prize in Artificial Intelligence. (2016). Loetud 5. aprill 2016 aadressil http://www.loebner.net/Prizef/loebner-prize.html

Loebner *Prize.* (2016) Loetud 5. aprill 2016 aadressil http://www.aisb.org.uk/events/loebner-prize

Perreau, E., Morton, D. (2016). *Program O Version 2.4.8*. Loetud 7. aprill 2016 aadressil https://github.com/Program-O/Program-O

Program O issues. (2016). Loetud 7. aprill 2016 aadressil https://github.com/Program-O/Program-O/issues

AIML *Implementations*. (kuupäev puudub). Loetud 9. aprill 2016 aadressil http://www.alicebot.org/downloads/programs.html

Program O API Status. (2016). Loetud 7. aprill aadressil http://program-o.com/chatbotapi

Program-*O Version 1*. (2009). Loetud 15. aprill 2016 aadressil https://sourceforge.net/projects/program-o/files/?source=navbar

Stanley, L. (2016). *What is Software Prototyping? - Definition, Models & Tools*. Loetud 16. aprill 2016 aadressil http://study.com/academy/lesson/what-is-software-prototyping-definition-models-tools.html

Eesti Keeletehnoloogia Riiklik Programm. (2016). *Kõnesünteesiliidesed*. Loetud 14. aprill 2016 aadressil https://www.keeletehnoloogia.ee/et/ekt-projektid/konesunteesiliidesed

Eesti Keeletehnoloogia Riiklik Programm. (2016). *Kõnetuvastus*. Loetud 14. aprill 2016 aadressil https://www.keeletehnoloogia.ee/et/ekt-projektid/konetuvastus

Eesti Keeletehnoloogia Riiklik Programm. (2016). *Tarkvara*. Loetud 14. aprill 2016 aadressil https://www.keeletehnoloogia.ee/et/ekt-projektid/konetuvastus/tarkvara

Eesti Keele Instituut. (2016). *Rakendused*. Loetud 14. aprill 2016 aadressil https://www.eki.ee/heli/index.php?option=com_content&view=article&id=5&Itemid=466

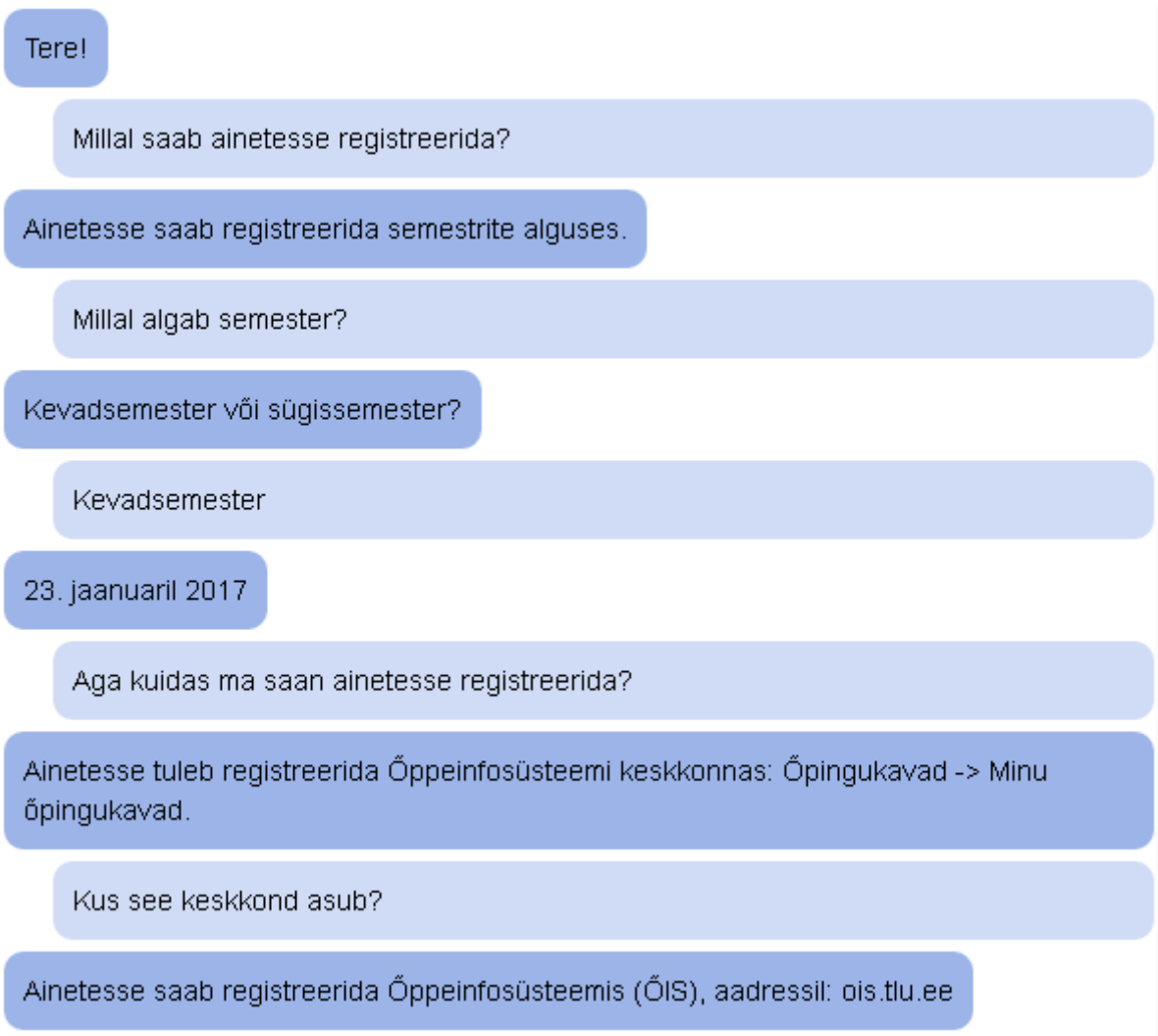
Akadeemiline kalender. (2016). *Tallinna Ülikooli akadeemiline kalender 2016/2017. õppeaastal.* Loetud 9. aprill aadressil http://www.tlu.ee/UserFiles/%C3%95pingud/akadeemiline_kalender_2016_2017.pdf

Wallace, R. (2003). *The elements of AIML style*. Loetud 11. aprill aadressil http://www.alicebot.org/style.pdf

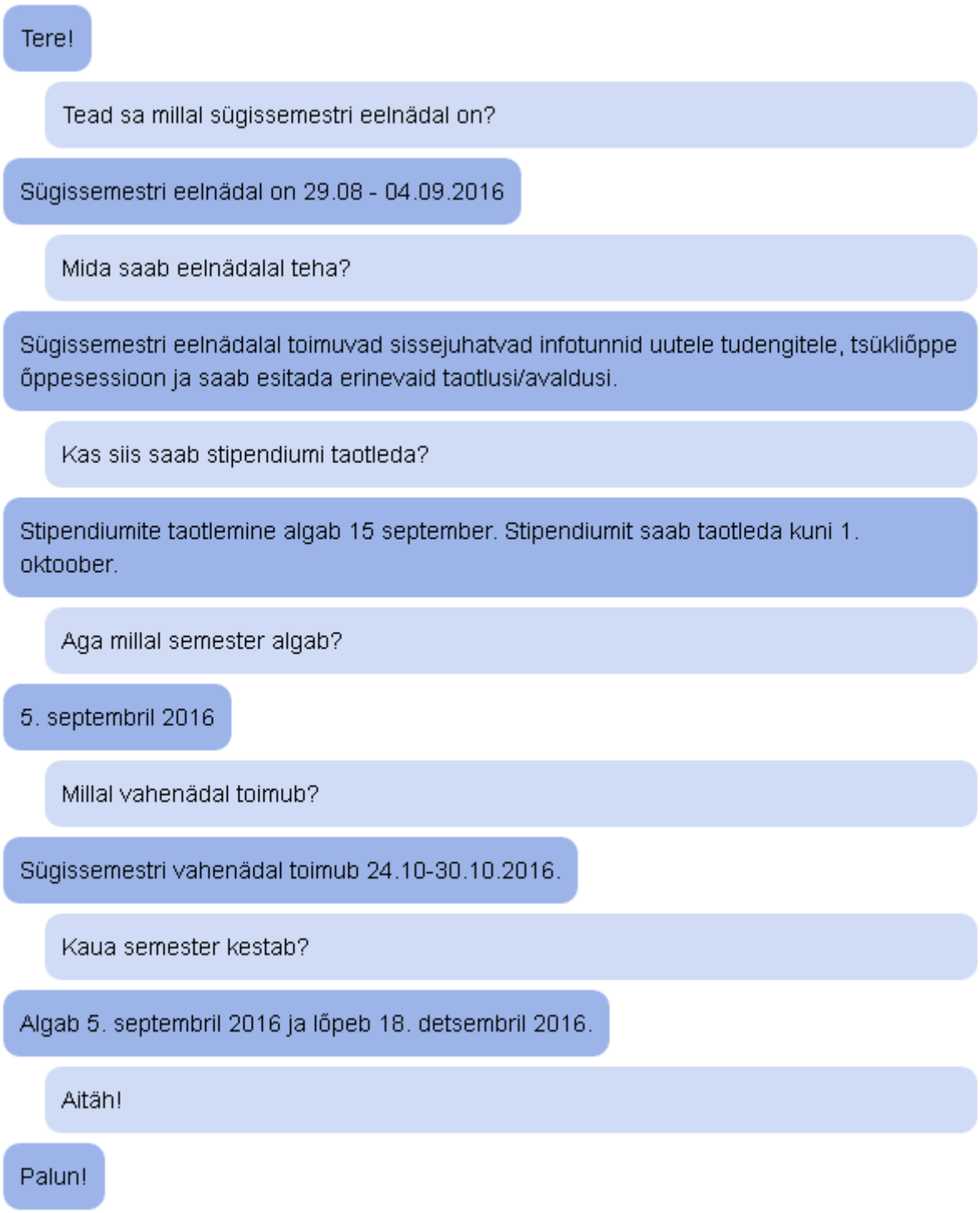
Marietto, M. Aguiar, R., Barbosa, G., Botelho, W., Pimentel, E., Franca, R., Silva, V. (2013). *Aritificial Intelligence Markup Language: A Brief Tutorial*. Loetud 5. aprill 2016 aadressil http://arxiv.org/ftp/arxiv/papers/1307/1307.3091.pdf

Lisad

Lisa 1. Vestlused

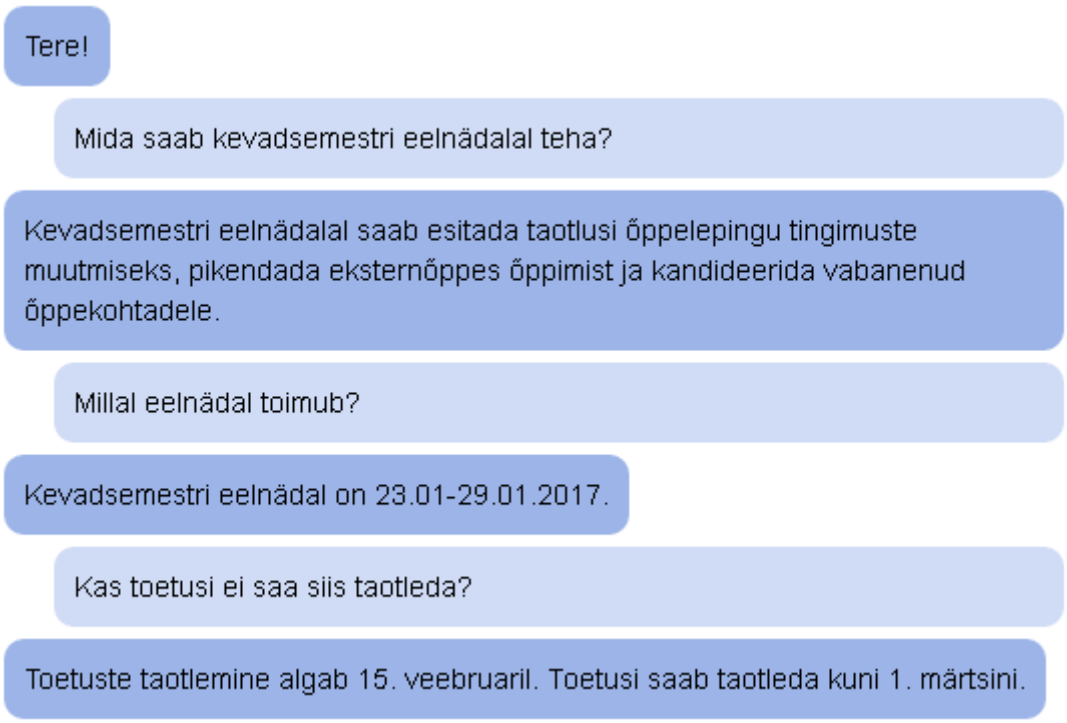


Lisa joonis 1. Vestlus prototüübiga.



Lisa joonis 2. Vestlus prototüübiga.

Lisa joonis 3. Vestlus prototüübiga.



Lisa joonis 4. Vestlus prototüübiga.