Raport ştiinţific şi tehnic in extenso

Program: *Parteneriate în domenii prioritare*

Proiect

Titlul: *Sistem de Asistenta pentru Clădiri Inteligente, controlat prin Voce şi Vorbire Naturala*

Acronim: *ANVSIB*

Data: *25.09.2017*

Etapa: *3/2016*

Activitatea / activitățile:

- *Activitatea 4.1. Integrarea sistemului de sinteza a vorbirii cu controlerul de voce*
- *Activitatea 4.2. Integrarea sistemului de recunoaştere a vorbirii cu controlerul de voce*
- *Activitatea 4.3. Scenarii de utilizare si configurare*
- *Activitatea 4.4. Integrarea hardware a infrastructurii KNX*
- *Activitatea 4.5. Evaluarea prototipului*
- *Activitatea 4.6. Diseminarea rezultatelor proiectului (partea II)*
- *Activitatea 4.7. Dezvoltarea unor modele acustice multilingve si pentru limba engleza (partea II)*
- *Activitatea 4.8. Integrarea algoritmului de căutare bazat pe DTW*
- *Activitatea 4.9. Implementarea infrastructurii si algoritmilor de detecție multilingva a termenilor*
- *Activitatea 4.10. Compararea si evaluarea performantelor tehnicilor implementate*
- *Activitatea 4.11. Crearea unui prototip de sistem de recunoaştere multilingva a vorbirii*
- *Activitatea 4.12. Crearea unui sistem de sinteza a vorbirii pentru limba romana*
- *Activitatea 4.13. Crearea unui sistem de sinteza a vorbirii pentru limba engleza*
- *Activitatea 4.14. Cercetarea si implementarea unui nou modul de prozodie*
- *Activitatea 4.15. Integrarea cu HTS*
- *Activitatea 4.16. Optimizarea si configurarea sistemului de sinteza a vorbirii*

Număr contract: *32 / 2014*

Acord de colaborare: *10677/24.06.2014 UPB, 1744/03.06.2014 IWAVE, 164/19.06.2014 RACAI*

Autoritatea contractantă: *Unitatea Executivă pentru Finanțarea Învățământului Superior, a Cercetării, Dezvoltării şi Inovării*

Conducător de proiect: *UPB*

Director de Proiect: Horia Cucu

Cuprins

1 Obiectivele activităților desfăşurate

În ultima etapă a proiectului ANVSIB, obiectivele UPB au fost a) crearea prototipului de sistem de recunoaştere multilingvă a vorbirii si b) integrarea algoritmului DTW de cautare a cuvintelor cheie intr-un sistem complet de detectie a termenilor vorbiti.

RACAI a avut ca obiectiv crearea, configurarea si optimizarea a doua modele funcționale de sinteză a vorbirii pentru limbile română, respectiv engleză şi integrarea lor într-un prototip multilingv de sinteză a vorbirii.

Obiectivele IWAVE au fost a) integrarea infrastructurii hardware de tip KNX cu controlerul de voce şi sistemele de recunoaştere, respectiv sinteză a vorbirii şi b) evaluarea prototipului final.

2 Perioada de desfăşurare şi personalul implicat

Nr.	Denumire activitate	Nume si prenume	Perioada
1.	*Activitatea 4.1. Integrarea sistemului de sinteza a vorbirii cu controlerul de voce*	*Ştefan Daniel Dumitrescu; Tiberiu Boroş Marian Bădulescu*	*noi 2016 – sept 2017*
2.	*Activitatea 4.2. Integrarea sistemului de recunoa tere a vorbirii cu controlerul de voce*	*Marian Bădulescu; Horia Cucu; Lucian Georgescu; Alexandru Caranica; Andi Buzo; Corneliu Burileanu; Florentina Mincă*	*noi 2016 – sept 2017*
3.	*Activitatea 4.3. Scenarii de utilizare si configurare*	*Marian Bădulescu; Horia Cucu; Lucian Georgescu; Alexandru Caranica; Andi Buzo; Corneliu Burileanu; Florentina Mincă; Ştefan Daniel Dumitrescu; Tiberiu Boroş*	*noi 2016 – sept 2017*
4.	*Activitatea 4.4. Integrarea hardware a infrastructurii KNX*	*Marian Bădulescu;*	*noi 2016 – sept 2017*
5.	*Activitatea 4.5. Evaluarea prototipului*	*Marian Bădulescu;*	*noi 2016 – sept 2017*
6.	*Activitatea 4.6. Diseminarea rezultatelor proiectului (partea II)*	*Horia Cucu; Lucian Georgescu; Alexandru Caranica; Andi Buzo; Ştefan Daniel Dumitrescu; Tiberiu Boroş*	*noi 2016 – sept 2017*
7.	*Activitatea 4.7. Dezvoltarea unor modele acustice multilingve si pentru limba engleza (partea II)*	*Horia Cucu; Lucian Georgescu; Alexandru Caranica; Andi Buzo; Corneliu Burileanu; Florentina Mincă*	*noi 2016 – sept 2017*
8.	*Activitatea 4.8. Integrarea algoritmului de căutare bazat pe DTW*	*Horia Cucu; Lucian Georgescu; Alexandru Caranica; Andi Buzo; Corneliu Burileanu; Florentina Mincă*	*noi 2016 – sept 2017*
9.	*Activitatea 4.9. Implementarea infrastructurii si algoritmilor de detec ie multilingva a termenilor*	*Horia Cucu; Lucian Georgescu; Alexandru Caranica; Andi Buzo; Corneliu Burileanu; Florentina Mincă*	*noi 2016 – sept 2017*
10.	*Activitatea 4.10. Compararea si evaluarea performantelor tehnicilor implementate*	*Horia Cucu; Lucian Georgescu; Alexandru Caranica; Andi Buzo; Corneliu Burileanu; Florentina Mincă*	*noi 2016 – sept 2017*
11.	*Activitatea 4.11. Crearea unui prototip de sistem de recunoa tere multilingva a vorbirii*	*Horia Cucu; Lucian Georgescu; Alexandru Caranica; Andi Buzo; Corneliu Burileanu; Florentina Mincă*	*noi 2016 – sept 2017*
12.	*Activitatea 4.12. Crearea unui sistem de sinteza a vorbirii pentru limba romana*	*Ştefan Daniel Dumitrescu; Tiberiu Boroş*	*noi 2016 – sept 2017*
13.	*Activitatea 4.13. Crearea unui sistem de sinteza a vorbirii pentru limba engleza*	*Ştefan Daniel Dumitrescu; Tiberiu Boroş*	*noi 2016 – sept 2017*
14.	*Activitatea 4.14. Cercetarea si implementarea unui nou modul de prozodie*	*Ştefan Daniel Dumitrescu; Tiberiu Boroş*	*noi 2016 – sept 2017*
15.	*Activitatea 4.15. Integrarea cu HTS*	*Ştefan Daniel Dumitrescu; Tiberiu Boroş*	*noi 2016 – sept 2017*
16.	*Activitatea 4.16. Optimizarea si configurarea sistemului de sinteza a vorbirii*	*Ştefan Daniel Dumitrescu; Tiberiu Boroş*	*noi 2016 – sept 2017*

3 Rezumat activități

3.1 Activitatea 4.1. Integrarea sistemului de sinteza a vorbirii cu controlerul de voce

Integrarea sistemului de sinteză a vorbirii cu controllerul de voce este prezentată în contextul întregului sistem, deoarece este parte integrantă.

Arhitectural, sistemul poate fi descris din mai multe puncte de vedere, însă în acest capitol vom descrie modul de interconectare al diferitelor componente şi module, cu accent mai puțin pe detalierea fiecăruia.

Figura de mai sus prezintă principalele componente ale sistemului. În partea stângă apar toate dispozitivele inteligente care vor fi controlate (lumini, aer condiționat, uşi, etc.). Serverul este punctul central, creierul casei. Totul va fi interconectat aici; tot aici are loc procesarea audio. Serverul poate să fie conectat la Internet, a.î. utilizatorul poate controla din exteriorul casei, spre exemplu, aerul condiționat. Interfețele de control (partea dreaptă) reprezintă puncte de interacțiune ale utilizatorului cu spațiul inteligent. În cadrul acestui proiect sunt disponibile două metode de control: dispozitivele mobile (telefon/tabletă/etc.) şi aşa numitele "dispozitive fixe". Aceste dispozitive sunt cutii mici (de dimensiunea unei doze electrice) care sunt conectate la server fie prin Wireless fie prin cablu Ethernet, şi au un microfon care ascultă permanent eventualele comenzi ale utilizatorului. De asemenea, sunt conectate la un set de difuzoare în aceeaşi cameră prin care comanda dată va fi confirmată.

Prezentăm în continuare fiecare componentă separat, cu accent pe modul în care modulele discută între ele (protocolul de comunicație folosit).

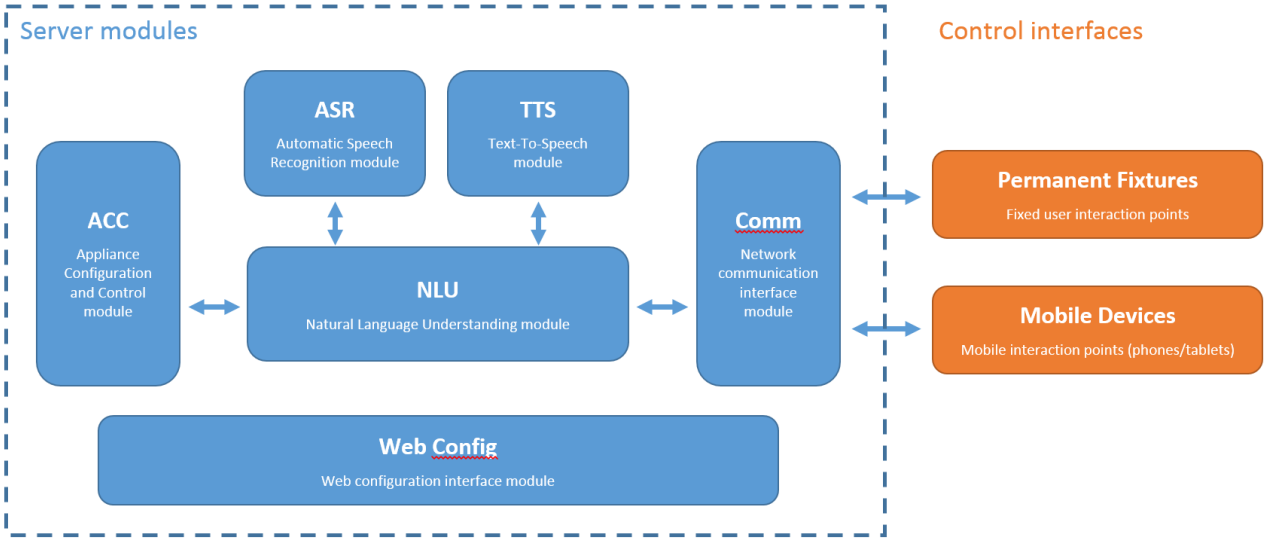

Diagrama de mai sus prezintă fiecare modul al sistemului. Serverul, din punct de vedere logic, are următoarele module:

- **Modulul NLU – Natural Language Understanding**

Acest modul este creierul spațiului inteligent. Primeşte şi trimite mesaje de la/către toate celelalte module. Sarcina principală este de a înțelege mesajele primite (mod text) pentru ca să le poată „traduce" mai departe către modului ACC (Appliance Configuration and Control).

- **Modului TTS – Text-to-Speech**

Aici are loc sinteza mesajelor text pentru a obține voce. Modulul primeşte un mesaj text, limba în care se doreşte sinteza (engleză sau română) si întoarce un stream wave audio direcționat către punctul de la care s-a primit mesajul text.

- **Modulul ASR – Automatic Speech Recognition**

În acest modul are loc procesul de recunoaştere de voce, primind un stream wave audio şi returnând un mesaj text. Este practic inversul procesului de sinteză din modulul precedent.

- **Modulul ACC – Appliance Configuration and Control module**

În acest modul are loc configurarea inițială a dispozitivelor existente în spațiul inteligent. Protocolul de comunicație folosit este KNX, interconectarea fiind efectuata prin TCP/IP (Ethernet sau Wireless acolo unde dispozitivul permite configurare).

- **Modulul Web**

Acest modul este de fapt un server standard Apache, având instalat un site care permite utilizatorului să configureze toate device-urile din casă, alături de metodele de apel către aceste device-uri (adică utilizatorul îşi poate defini comanda proprie de control asupra fiecărui dispozitiv).

Interfețele de control sunt fie dispozitive mobile fie fixe:

- **Dispozitive mobile**

Utilizatorul poate controla casa inteligentă folosind telefonul sau tableta proprie, sau orice alt dispozitiv inteligent care poate rula aplicații Android si are acces la server prin Wireless local sau prin conexiune 3G/4G la server.

- **Dispozitive fixe**

Un punct fix reprezintă un punct de interacțiune plasat în cameră. Acest dispozitiv ascuns, de exemplu, în perete într-o doză electrică sau direct aplicat pe perete, are un microfon care stă permanent pornit, ascultând eventualele comenzi. Acest dispozitiv fix ascultă un cuvânt cheie prin care se începe comanda către casă, şi, în funcție de configurație, fie procesează semnalul audio local (adică efectuează procesul de ASR local trimițând către server doar textul comenzii) fie trimite fluxul audio către server, serverul ocupându-se mai departe de procesare. Primeşte înapoi de la server confirmarea comenzii. Dispozitivul în sine este un sistem x86 sau ARM tip System-On-A-Chip (ex: Raspberry Pi) de dimensiuni reduse.

3.2 Activitatea 4.2. Integrarea sistemului de recunoaştere a vorbirii cu controlerul de voce

Modulul ASR (Automatic Speech Recognition), alături de modulul TTS prezentat în activitatea anterioară, a fost prezentat în contextul întregului sistem (ca mod de legătură), nemai fiind necesară o descriere în acest punct. Însă, pentru a întelege modul în care ASR-ul şi TTS-ul sunt integrate cu sistemul, considerăm oportună prezentarea unui exemplu de comunicație între aceste module, alături de toate celelalte module prezente pe server.

Exemplu de comunicație între module.

Să presupunem că avem un scenariu simplu în care utilizatorul doreşte *închiderea luminilor din bucătărie.* Vom exemplifica în paralel pentru ambele tipuri de interfețe de control.

- **Pasul 1**. Utilizatorul dă comanda „Casandra închide lumina în bucătărie" pe telefonul său mobil. Mobilul înregistrează comanda şi o trimite mai departe către NLU sub formă de audio raw (fişier wav). Dacă comanda a fost trimisă prin intermediul unui dispozitiv fix şi dacă acel dispozitiv fix rulează procesul de ASR local, audio-ul va fi transformat direct pe dispozitivul fix în mesaj text, către server trimițându-se doar mesajul text, sărind peste pasul 2.
- **Pasul 2**. Audio-ul este trimis de către NLU către modulul de ASR. De acolo primeşte înapoi un mesaj text.
- **Pasul 3**. Modulul NLU analizează mesajul şi îl transformă într-o comandă efectivă către un dispozitiv inteligent din casă. În exemplul curent, mesajul transformat arată astfel: {target_appliance="kitchen_lights", action="off"}.
- **Pasul 4**. Modulul ACC efectuează comanda, şi anume închide luminile din bucătărie. În acest pas, este posibil ca dispozitivul către care se trimite comanda să ofere un feedback către utilizator de tipul: lumina a fost închisă, sau Uşa a fost încuiată.
- **Pasul 5**. Daca modulul NLU aşteaptă un feedback, în funcție de statusul primit va trimite către dispozitivul de interacțiune un mesaj care poate fi fie mesajul standard "comandă confirmată" sau un mesaj corespunzător comenzii utilizatorului (ex: "volumul a fost setat la 60%").
- **Pasul 6.** Dispozitivul de interacțiune (fie mobil fie fix) va primi mesajul text şi îl va sintetiza prin trimiterea directă catre modulul TTS a mesajului. Modulul TTS trimite direct înapoi audio-ul raw care este redat fie de către telefonul mobil, fie prin difuzoarele la care este legat dispozitivul fix.

3.3 Activitatea 4.3. Scenarii de utilizare si configurare

Sistemul NLU este bazat pe scenarii. În orice moment, utilizatorul poate modifica configurația sistemului (Cassandrei) prin adăugarea, modificarea sau ştergerea de scenarii de utilizare.

Un scenariu este definit de un nume şi de trei componente:

1. O lista de exemple de propoziții în care fiecare parametru este precedat de un semn special ($). De exemplu: "Setează temperatura la * grade" unde * este un parametru variabil (utilizatorul poate spune "zece", "două zeci şi şapte", etc.
2. O listă de acțiuni pe care o va efectua sistemul în urma identificării comenzii definite anterior. Acțiunile sunt definite în format JSON astfel: a. id-ul dispozitivului inteligent (ex: o anume unitate A/C, un set de becuri, o anumită uşă, etc.); b. Parametrii comenzii – o structură fără format fix JSON care îi spune sistemului ce parametrii să trimită mai departe dispozitivului inteligent. (ex: aprinde lumina on=1, stinge lumina off=0, setează A/C la X grade, etc.) Valorile acestor parametri pot fi fie constante, fie variabile predefinite, sau valori efective extrase din text (adică string-uri, numere, etc).
3. Feedback. După procesare, JSON-ul definit aici este pasat dispozitivului de control care a inițiat sesiunea de comenzi, fiind folosit pentru a transmite informație utilizatorului. Acest JSON are de asemenea două atribute: a. Friendly_response – un răspuns text care, dacă există, va fi sintetizat şi redat utilizatorului; răspunsul poate conține variabile predefinite sau extrase din comanda inițială, sau generate de dispozitiv. (exemplu: temperatura efectivă din cameră ca răspuns oferit de aparatul de aer condiționat la întrebarea utilizatorului asupra temperaturii ambientale). b. Launch_intent – o structură care informează dispozitivul de interacțiune (telefonul mobil în acest caz) ce aplicație trebuie să lanseze (tip Android Intent) cu un anumit set de parametrii. Casandra poate genera acțiuni pentru previzualizarea de imagini, audio şi video pe telefoane mobile.

Metodologia pentru identificarea scenariilor si extragerea de parametrii este efectuata într-o serie de paşi secvenţiali.

a. Scenariul este identificat utilizând un arbore de decizie tip ID3. Există si posibilitatea de a utiliza o reţea convoluţională special creată, aceasta însă necesitând o serie de date lingvistice pe baza cărora se generează o codare a cuvintelor (folosind word2vec). În producţie, se foloseşte arborele de decizie pentru a identifica scenariul, deoarece nu este necesară nicio altă resursă lingvistică care să scadă din universalitatea metodei de identificare.
   Odată definite toate scenariile, se extrag propoziţiile pentru fiecare scenariu. Generarea arborelui de decizie se efectuează extrăgând într-un format bag-of-words fiecare cuvânt din fiecare comandă, având asociat scenariul efectiv. Aşadar, pentru fiecare scenariu avem cel puţin o propoziţie (de obicei minim 3-4). Algoritmul de training analizează fiecare propoziţie (comandă) în parte, generând un arbore. Arborele este redus ca dimensiune pentru a garanta un anumit nivel de universalitate.
b. Extragerea parametrilor din comandă este efectuată tot cu un arbore de decizie tip ID3. În primul rând trebuie identificată poziţia parametrilor, acest lucru fiind efectuat în doi paşi. În primul pas este identificată poziţia de start (indicele cuvântului care începe parametrul – un parametru poate avea mai mult de un cuvânt, de exemplu "temperatura la două zeci şi şapte de grade" are un singur parametru de 4 cuvinte) folosind un arbore antrenat pentru a prezice, pe fiecare cuvânt, fie clasa Start (ex: cuvântul "două") fie Null (toate celelalte cuvinte). Odată identificat un start de parametru, se foloseşte un alt arbore de decizie, antrenat pentru a prezice Stop (ex: cuvântul "şapte") sau Null. Parametrul este cel între Start-Stop, inclusiv. Odată marcat un cuvânt cu Stop, se porneşte căutarea cu arborele care prezice Start, deoarece este posibil ca o comandă sa aibă mai mulţi parametri. Contextul folosit pentru arbori este de 4 cuvinte înaintea cuvântului analizat pentru arborele Start, respectiv 4 cuvinte după pentru arborele Stop.
   Odată identificaţi parametrii, trebuie găsit tipul fiecăruia. Acest lucru se face recurgând la un alt arbore de decizie, care stabileşte dacă parametrul este valoare numerică, string, predefinit, etc.

In continuare prezentăm câteva scenarii practice de utilizare.

Scenariul principal: utilizatorul controlează aparatele electrocasnice folosind vocea. Există mai multe sarcini predefinite cum ar fi controlul multimedia, iluminare, climatizare sau securitate. De exemplu: "Casandra, porneşte sistemul de climatizare" (en. Casandra, turn on the air-conditioning system), "Casandra, aprinde toate luminile" (en. Casandra, turn on all the lights), "Casandra, activează stropitorile" (en. Casandra, start the irrigation system), "Casandra, măreşte temperatura cu trei grade în sufragerie" (en. Casandra, raise the living room temperature by three degrees). În acest scenariu se încadrează toate interacţiunile cu dispozitivele inteligente din casă.

Scenariul 2: reprezintă un sistem standard de interogare (întrebare-răspuns sau QA), în care utilizatorul îi poate adresa Casandrei diverse întrebări (de ex. "cum este vremea") la care sistemul va răspunde folosind o bază de cunoştinţe (KB), similară cu ce ar răspunde Google Now, Stiri sau Amazon Echo. Baza de cunoştinţe (KB) a fost construită folosind Wikipedia pentru limba romană. Acest modul este de tip proof-of-concept, nefiind prevăzut în cadrul proiectului dar fiind implementat pentru a extinde funcţionalitatea sistemului în afara controlului direct al dispozitivelor inteligente.

Scenariul 3: este un set de întrebări scurte / răspunsuri. Acest set de Q/A conţine un joc care se numeşte "psihologul". Acesta permite sistemului să efectueze asociaţii de cuvinte într-un mod similar în care un psiholog ar cere unui pacient să o facă. Acest joc s-a dovedit a fi foarte atrăgător pentru utilizatorii din grupul nostru de testare, în primul rând pentru ca a făcut sistemul sa pară mai "asemănător unui om". Idea de bază este că, folosind Wikipedia si word2vec, am extras seturi de cuvinte similare. Rostind un cuvânt, "psihologul" va şti să rostească un alt cuvânt cu aproape acelaşi sens (dpdv. Semantic), jocul continuând în runde.

3.4 Activitatea 4.4. Integrarea hardware a infrastructurii KNX

Asociaţia KNX este creatoarea şi proprietara tehnologiei KNX - STANDARD mondial pentru toate aplicaţiile de control pentru case şi clădiri, variind de la controlul iluminatului şi al jaluzelelor până la variate sisteme de securitate, încălzire, ventilaţie, aer condiţionat, monitorizare alarme, controlul instalaţiilor sanitare, managementul energiei, contorizare, precum şi al dispozitivelor de uz casnic, audio/video şi multe altele. Tehnologia KNX poate fi utilizată atât în clădiri noi cât şi în cele deja existenţe. KNX este standardul global pentru controlul clădirilor cu o singură platformă software, independentă de un producător KNX, pentru design şi punere în funcţiune (ETS), cu o gamă completă de medii de transmisie disponibile (TP, PL, RF şi IP), precum şi cu o gamă completă de moduri de configurare (Modul System şi Modul Easy). KNX este aprobat ca Standard European (CENELEC EN 50090 şi CEN EN 13321-1) şi ca Standard Internaţional (ISO/IEC 14543-3). Pentru a asigura transferul datelor între toate componentele de management al clădirii, este necesar un sistem care să rezolve problema comunicării cu dispozitivele izolate, asigurându-se că toate componentele comunică într-un limbaj comun unic: pe scurt, un sistem precum Magistrala KNX, independent de producător şi de domeniul de aplicaţie. Acest standard se bazează pe mai mult de 23 de ani de experienţă pe piaţă, printre altele, cu sistemele predecesoare KNX: EIB, EHS şi BatiBUS. În mediul KNX toate dispozitivele sunt conectate la acelaşi BUS (pereche torsadată, RF, linii electrice sau IP/Ethernet) cu posibilitatea de a comunica între ele. Dispozitivele KNX pot fi: senzori sau elemente de acţionare, necesare controlului diferitelor sisteme ale clădirii,

precum: iluminat, jaluzele/storuri, sisteme de securitate, managementul energiei, încălzire, ventilație şi sisteme de aer condiționat, sisteme de semnalizare şi monitorizare, interfețe cu sistemele de service şi control al clădirii, control de la distanță, contorizare, control audio/video, electrocasnice, etc. Toate aceste funcții sunt controlate, monitorizate şi semnalizate prin intermediul unui sistem unitar fără a fi necesare centre de control suplimentare. (descriere preluată de pe www.knx.ro)

Deoarece în standardul KNX nu există un controller (fiecare echipament este independent şi ascultă pe bus), practic efortul de integrare a fost majoritar dozat in configurația software a echipamentelor. Din punct de vedere hardware tot ce trebuie făcut este conectarea lor în cadrul unei rețele. Există mai multe metode suportate de KNX pentru interconectare (în funcție de interfața echipamentului):

1. KNX TP: Twisted Pair, adică o pereche de fire de cupru subțiri, este mediu vechi de comunicare, preluat din standardul EIB, viteza 9600bps, oferă contact punct-la-punct.
2. KNX RF: Radio Frequency, adica conexiune prin radio. Pachetele se transmit pe distanță mică, frecvența de 868MHz, puterea maximă permisă fiind de 25mW.
3. KNX PL: Power Line, conexiune direct pe linia de curent, viteză de 1200bps.
4. KNX IP: mesajele sunt transmise pe rețeaua de date TCP/IP. Marea parte a dispozitivelor inteligente oferă această interfață.

Echipamentele conectate/controlate de I-WAVE în cadrul proiectului au fost: serie de lumini, dispozitiv audio multimedia inteligent (conectat la internet), lumini colorate, sistem climatizare automată, sistem de securitate cu mai multe partiții activabile individual. Toate aceste echipamente au fost conectate, configurate şi controlate de către Cassandra. Unele dintre ele se leaga in KNX direct (daca au interfata KNX), în timp ce marea majoritate cu Comfortclick prin HTTP API; unele comunica prin infraroşu IR (ex aerul conditionat individual).

Comfortclick integreaza mai multe protocoale (HUE, KNX, MODBUS, HTTP API, IR) si le pune la dispozitie pe portul 84 prin JSON .

Efortul de configurare hardware a constat în adăugarea fiecărui dispozitiv în Comfortclick, identificarea printr-un ID unic şi preluarea acelui ID şi introducerea în interfața de configurare a Cassandrei. În acea interfață au trebuit definite acțiunile pentru fiecare dispozitiv. De exemplu, pentru lumini sunt necesari doi parametri: starea luminii (aprins/stins – variabila booleană) şi poziția luminii (adică locația ei, de exemplu bucătărie este lumina numărul 1, sufragerie numărul 2, etc. – variabilă număr întreg). Pentru luminile colorate mai sunt necesari încă doi parametri, şi anume culoarea (variabilă tip RGB) şi intensitatea (ex: 100%, 60%, etc. – variabilă număr întreg). În final, trebuie setate un număr de propoziții parametrizabile pe care utilizatorul le va folosi pentru a efectua o acțiune. De exemplu, pentru a aprinde lumina, utilizatorul va defini propoziția „aprinde lumina în LOC" unde LOC este oricare din locațiile anterior definite, de tipul bucătărie = 1, sufragerie = 2, etc, iar acțiunea este definită prin a pasa o valoare de tip bool "value=true" dispozitivului bec cu id-ul LOC.

Aşadar, această activiate a presupus studierea fiecărui dispozitiv în parte; conectarea fizică a fiecărui dispozitiv în funcție de interfața de comunicație; asignarea în Comfortclick a unui ID unic; configurarea în interfața de management a Cassandrei a fiecărui dispozitiv şi definirea parametrilor în JSON-ul propriu alături de acțiunea propriuzisă.

3.5 Activitatea 4.5. Evaluarea prototipului

Serverul este "creierul" casei. În sine este un calculator de dimensiuni reduse. Serverul se ocupă cu toată partea de procesare efectivă, aşadar are nevoie de resurse de calcul. Testele în cadrul proiectului s-au desfăşurat pe un calculator cu procesor Intel i3 cu 4 GB RAM si un SSD pentru stocare. Dimensiunea SSD-ului nu este relevantă deoarece nu stocăm practic nimic în afară de sistemul de operare reprezentat de un Linux, alături de modulele software care în sine au o dimensiune nesemnificativă comparativ cu spațiul oferit de orice SSD modern, adică cel puțin 100GB). Serverul funcționează separat, nefiind nevoie de acces la Internet. Daca totuşi este conectat, serverul poate primi comenzi de la dispozitive mobile din afara razei Wireless sau a rețelei locale.

Controllerul de voce, reprezentat prin modulul NLU (Natural Language Understanding) este punctul central al serverului, deoarece leagă toate celelalte module între ele. Misiunea acestui modul este de a traduce mesajele generate de ASR in mesaje către modulul de control al dispozitivelor inteligente din casă.

Evaluarea prototipului consta în implementarea unor funcționalități practice, bazare pe dispozitive atât reale cât şi virtuale. Astfel, au fost implementate următoarele scenarii (bazate pe tipul 1 de scenariu prezentat în secțiunea precedentă – * reprezintă valoarea dorită, LOC reprezintă o locație predefinită de tipul "bucătărie","dormitor","baie",etc.):

```
HVAC - Read temperature
        casandra care este temperatura din LOC
        casandra câte grade sunt în LOC
        casandra citeşte temperatura din LOC

HVAC - Set absolute temperature
        casandra setează temperatura din LOC la * de
grade
        casandra schimbă temperatura din LOC la * de
```

```
Lights - Set decrease intensity
        casandra micşorează luminozitatea din LOC la *
la sută
        casandra micşorează luminozitatea din LOC până
la * la sută
        casandra scade luminozitatea din LOC la * la
sută
        casandra micşorează intensitatea luminii din
LOC până la * la sută
```

```
grade
        casandra dă aerul condiţionat pe * grade
        casandra porneşte aerul condiţionat pe * grade
        casandra schimbă aerul condiţionat pe * grade
        casandra pune aerul condiţionat pe * grade
        casandra setează aerul condiţionat pe * grade

HVAC - Set increase temperature
        casandra creşte temperatura din LOC cu * grade
        casandra setează aerul condiţionat mai cald
        casandra dă aerul condiţionat mai cald

HVAC - Set decrease temperature
        casandra scade temperatura din LOC cu * grade
        casandra setează aerul condiţionat mai rece
        casandra dă aerul condiţionat mai rece

HVAC - Set windows off
        casandra închide ferestrele din LOC
        casandra închide geamurile din LOC

HVAC - Set windows on
        casandra deschide ferestrele din LOC
        casandra deschide geamurile din LOC

HVAC - Set window blinds off
        casandra închide obloanele din LOC
        casandra închide jaluzelele din LOC

HVAC - Set window blinds on
        casandra deschide obloanele din LOC
        casandra deschide jaluzelele din LOC

Lights - Set on
        casandra aprinde lumina din *
        casandra porneşte lumina din *
        casandra deschide lumina din *

Lights - Set off
        casandra stinge lumina din *
        casandra opreşte lumina din *
        casandra închide lumina din *

Lights - Set on all
        casandra aprinde toate luminile
        casandra porneşte toate luminile
        casandra deschide toate luminile

Lights - Set off all
        casandra opreşte toate luminile
        casandra închide toate luminile
        casandra stinge toate luminile
Lights - Set color
        casandra schimbă culoarea luminii din * în *
        casandra aprinde lumina * în *
        casandra porneşte lumina * în *
        casandra deschide lumina * în *

Lights - Set absolute intensity
        casandra setează luminozitatea din LOC la * la
sută
        casandra schimbă luminozitatea din LOC la * la
sută
        casandra setează intensitatea luminii din LOC
la * la sută
        casandra schimbă intensitatea luminii din LOC
la * la sută

Lights - Set increase intensity
        casandra măreşte luminozitatea din LOC la * la
sută
        casandra măreşte luminozitatea din LOC până la
* la sută
        casandra creşte luminozitatea din LOC la * la
sută
        casandra creşte luminozitatea din LOC până la
* la sută
```

```
Security - Call police
        casandra sună la poliţie
        casandra cheamă poliţia
        casandra sună poliţia

Security - Call ambulance
        casandra sună la salvare
        casandra sună la ambulanţă
        casandra cheamă salvarea
        casandra cheamă ambulanţa
        casandra sună salvarea
        casandra sună ambulanţa

Security - Call firefighters
        casandra sună la pompieri
        casandra cheamă pompierii
        casandra sună pompierii

Security - Set efraction state on
        casandra armează centrala de alarmă
        casandra armează partiţia cu numărul *
        casandra armează partiţia *
        casandra activează centrala de alarmă
        casandra activează partiţia cu numărul *
        casandra activează partiţia *

Security - Set efraction state off
        casandra dezarmează centrala de alarmă
        casandra dezarmează partiţia cu numărul *
        casandra dezarmează partiţia *
        casandra dezactivează centrala de alarmă
        casandra dezactivează partiţia cu numărul *
        casandra dezactivează partiţia *

Security - Get efraction state
        casandra care este statusul partiţiei *
        casandra ce status are partiţia *
        casandra cum este partiţia *
        casandra status partiţie *
        casandra statusul partiţiei *


Multimedia - Set absolute sound intensity
        casandra setează volumul televizorului la * la
sută
        casandra schimbă volumul televizorului la * la
sută
        casandra setează volumul muzicii la * la sută
        casandra schimbă volumul muzicii la * la sută

Multimedia - Set increase sound intensity
        casandra măreşte volumul televizorului
        casandra creşte volumul televizorului

Multimedia - Set decrease sound intensity
        casandra micşorează volumul televizorului
        casandra scade volumul televizorului

Multimedia - Set on
        casandra porneşte muzica
        casandra porneşte radio
        casandra porneşte radioul
        casandra porneşte televizorul

Multimedia - Set off
        casandra opreşte muzica
        casandra opreşte radio
        casandra opreşte radioul
        casandra opreşte televizorul

Multimedia - Set radio station
        casandra vreau să ascult radio *
```

La fiecare scenariu Casandra va efectua ac iunea dorită i va confirma respectiva ac iune printr-un mesaj sintetizat. Prototipul a fost testat pentru fiecare din scenariile de mai sus, confirmându-se func ionarea corectă.

3.6 Activitatea 4.6. Diseminarea rezultatelor proiectului (partea II)

Pentru această etapă finală, activitatea de diseminare a RACAI a rezultatelor proiectului s-a efectuat prin participarea la 2 conferinţe unde au fost prezentate 3 articole:

Conferința SPED17: The 9th Conference on Speech Technology and Human-Computer Dialogue, Iulie 6-9, 2017 – Bucuresti, Romania

- Articolul 1: A "Small-Data" - Driven Approach to Dialogue Systems for Natural Language Human Computer Interaction , Tiberiu Boroş, Ştefan Daniel Dumitrescu. Acest articol prezinta modul de configurare al Cassandrei din punct de vedere al scenariilor (interacțiune om-casa inteligenta).
- Articolul 2: Cassandra Smart-Home System Description, Ştefan Daniel Dumitrescu. Acest articol prezinta o descriere a Cassandrei din punct de vedere al arhitecturii logice şi al implementării modulelor.

Conferința ACL (Association for Computational Linguistics), workshop CONLL (The SIGNLL Conference on Computational Natural Language Learning), Vancouver, Canada, 3-4 August 2017

- Articol 1: RACAI's Natural Language Processing pipeline for Universal Dependencies, Stefan Daniel Dumitrescu, Tiberiu Boroş and Dan Tufiş. Acest articol prezinta preprocesatorul de text prezent în Casandra şi modul in care a fost extins la un număr arbitrar de limbi (unul din țintele acestui proiect), utilizând resursele lingvistice puse la dispoziție in cadrul competiției CONLL 2017.

Pentru această etapă finală, activitatea de diseminare a UPB a rezultatelor proiectului s-a efectuat prin participarea la o conferință ISI şi transmiterea spre publicare a unui articol de jurnal ISI:

- Alexandru Caranica, Horia Cucu, Corneliu Burileanu, François Portet, Michel Vacher, "Speech Recognition Results for Voice-controlled Assistive Applications," in the Proceedings of the 9th Conference on Speech Technology and Human-Computer Dialogue (SpeD), Bucharest, 2017, 8p, ISBN 978-1-5090-6496-0;
- Alexandru Caranica, Horia Cucu, Andi Buzo, Corneliu Burileanu, "Survey on Multilingual Spoken Term Detection," in Romanian Journal of Information Science and Technology (trimisă spre publicare).

3.7 Activitatea 4.7. Dezvoltarea unor modele acustice multilingve si pentru limba engleza (partea II)

În cadrul acestei activități, au fost realizate experimente ce au avut ca scop crearea unor modele acustice, atât pentru limba engleză, cât şi pentru limba română. Corpusurile audio au fost împărțite în seturi de antrenare şi de evaluare, fiind utilizate modele de limbă probabilistice de tip n-gram. Pentru antrenarea modelelor s-a folosit atât utilitarul Kaldi cât şi Sphinx. Sphinx permite parametrizarea semnalului vocal şi extragerea coeficienților de tip MFCC, PLP sau LPC, în plus grupul nostru integrând şi parametrii robuşti la zgomot, de tip PNCC, pentru care am prezentat rezultatele în raportul anterior. Modelarea acustică folosind acest toolkit presupune crearea unor sisteme statistice de tip HMM-GMM.

În plus, față de Sphinx, Kaldi oferă antrenarea pe baza de rețele neuronale, ceea ce a oferit un spor de performanță pentru anumite modele, după cum am prezentat anterior. Avantajul major este reprezentat de existenta algoritmilor de antrenare ce folosesc rețele neuronale adânci (DNN), cu posibilitatea utilizării unui cluster de stații de lucru, echipate cu GPU-uri Tesla. În urma testelor, s-a observat faptul că această platformă de tip DNN oferă o acuratețe mai bună în comparație cu sistemele HMM-GMM din Sphinx.

În continuare, în cadrul grupului, am continuat achiziția de resurse lingvistice, în special pentru limba engleză. O nouă bază de date audio, intitulată homeAuto4, a fost achiziționată, în cadrul grupului SpeeD, pentru evaluarea performanțelor sistemului multilingv, în limba engleză. Un set de 154 de transcrieri, în limba engleză, au fost realizate, ce reprezintă comenzi ce pot fi înțelese şi interpretate de sistemul de recunoaştere a vorbirii. Un număr de 11 vorbitori de limba engleză, non-nativi, au realizat înregistrări audio, folosind noua versiune a aplicației "Speech Recorder", disponibilă la adresa https://speech-recorder.speed.pub.ro (Figura 1).

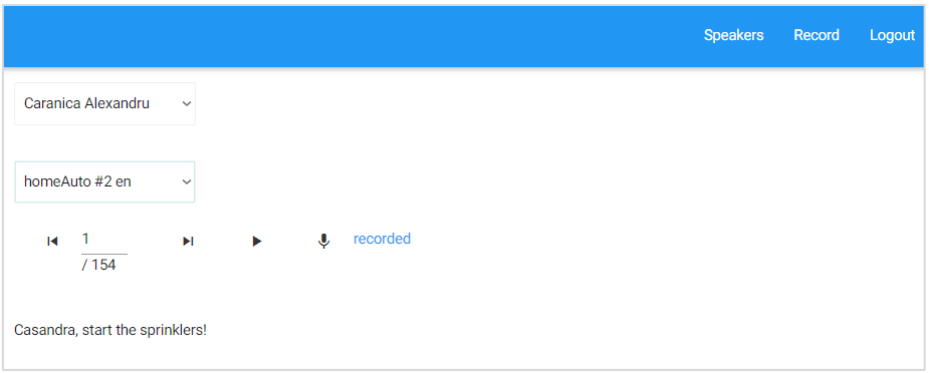

Figura 1 Noua versiune a aplicației Speech Recorder, a grupului SpeeD

Noul corpus de evaluare totalizează un număr de aproximativ 2 ore şi 10 minute de înregistrări, în limba engleză, înregistrările fiind realizate de către studenți şi personal al grupului de cercetare, cu vârste cuprinse între 24 şi

35 de ani, procentul de vorbitori feminini fiind de aproximativ 27%. Aceste informații, sumarizate, se regăsesc în Tabelul 1.

	Evaluare
Număr conversații	154
Durata totală	2 h 10 m
Vorbitori masculini	~95m
Vorbitori feminini	~35m
Durata medie	11 m 8 s
Număr de vorbitori unici	11
Vârsta	24-35 ani

Tabelul 1 Corpusul de evaluare homeAuto 4 achiziționat

Pentru realizarea modelului acustic, în limba engleză, s-a folosit corpusul de vorbire TED-LIUM, versiunea 1 [Rousseau, 2012], ce conține înregistrări audio şi transcrierile aferente de la conferințele Ted (https://www.ted.com). Corpusul a fost împărțit în 3 seturi distincte ale căror caracteristici se regăsesc în Tabelul 2.

	Antrenare	**Dezvoltare**	**Testare**
Număr de conversații	774	19	11
Durata totală • **Vorbitori masculini** • **Vorbitori feminini**	118 h 4 m 48 s 81 h 53 m 7s 36 h 11 m 41 s	4 h 12 m 55 s 3 h 13 m 57 s 58 m 58 s	3 h 4 m 5 s 2 h 21 m 35 s 42 m 29 s
Durata medie	9 m 9 s	13 m 18 s	16 m 43 s
Număr de vorbitori unici	666	19	11

Tabelul 2 Corpusul TED-LIUM

Construirea unui model multilingv RO-EN s-a realizat în CMU-Sphinx, prin antrenarea unui sistem folosind corpusurile de antrenare în limba engleză şi cele din limba română.

Pentru realizarea şi evaluarea modelului acustic în limba română s-a folosit un corpus ce conține vorbire citită în condiții de liniște, dar şi înregistrări ale unor emisiuni de televiziune şi radio, unele dintre acestea fiind afectate de diverse zgomote de fundal (muzică, râsete, etc.). Corpusul a fost împărțit conform Tabelul 3. Fonemele au fost antrenate în comun, pentru limba engleză şi română la un loc.

Set	**Subset**	**Durata**		**Conținut**
Antrenare	Omega train	94 h 46 m 14 s	225 h 31 m 31 s	Vorbire citita, fără zgomot
	TV train1 (TVDatabase)	27 h 27 m 21 s		Vorbire spontană, uneori cu zgomot
	TV train2 (TVautomaticAquisition)	103 h 17 m 56 s		Vorbire spontană, fără zgomot
Testare	Omega test	5 h 29 m 6 s	8 h 58 m 9 s	Vorbire citită, fără zgomot
	TV test (News all)	3 h 29 m 3 s		Vorbire spontană, uneori cu zgomot

Tabelul 3 Corpusul audio în limba română

Evaluarea modelului acustic multilingv RO-EN, în Sphinx, s-a realizat pe bazele de date homeAuto2, homeAuto3 şi homeAuto4.

Suplimentar, gramatica cu stări finite folosită în decodarea sistemului prototip conține şi un model de tip ”garbage”, pentru rejecția cuvintele din afara vocabularului, scopul fiind de a identifica comenzile greşite, sau propozițiile ce nu conțin comenzi. Astfel, în momentul în care o comandă nu este recunoscută, sistemul poate fi programat în a sintetiza un mesaj audio, prin care utilizatorului i se cere rostirea unei comenzi corecte. Figura 2 evidențiază o porțiune din gramatica în limba română, ce conține această regulă de tip ”garbage”.

```
<numar_ac> = zece | unsprezece | doisprezece | douasprezece | treisprezece | paisprezece | cincisprezece | saisprezece
| saptesprezece | optsprezece | nouasprezece | douăzeci | douăzeci_şi_unu | douăzeci_şi_doi | douăzeci_şi_trei |
douăzeci_şi_patru | douăzeci_şi_cinci | douăzeci_şi_şase | douăzeci_şi_şapte | douăzeci_şi_opt | douăzeci_şi_nouă |
treizeci | treizeci_şi_unu | treizeci_şi_doi | treizeci_şi_trei | treizeci_şi_patru | treizeci_şi_cinci |
treizeci_şi_şase | treizeci_şi_şapte | treizeci_şi_opt | treizeci_şi_nouă | patruzeci;


//-------------------------------------------------------------------------------------------------------------------
<din_in_camere> = (din | în) (bucătărie | baie | dormitor | sufragerie | cameră);


//--------------------
public <comandaCasa> = (/0.0001/garbage*) | (/0.999/casandra (<securitate> | <multimedia> | <hvac> | <electric>));
```

Figura 2 Gramatica cu stări finite, în limba română, ce conține ”garbage” model.

Bibliografie

[Rousseau, 2012] A. Rousseau, P. Deléglise, and Y. Estève, "TED-LIUM: an automatic speech recognition dedicated corpus", in Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12), May 2012.

3.8 Activitatea 4.8. Integrarea algoritmului de căutare bazat pe DTW

Sistemul propus de căutare directă, în cadrul fişierelor audio, foloseşte un model acustic multilingv cu un algoritm scalabil de tip Dynamic Time Warping (DTW). Pentru a încerca rezolvarea problemei căutării direct în cadrul fişierelor audio, fără a cunoaşte limba, am folosit un sistem de indexare cu o recunoaştere de fonem multilingvă. De asemenea, am experimentat cu mai multe tipuri de caracteristici, pentru a vedea care se comportă mai bine şi în ce condiţii, pentru această sarcină.

Implementarea finală a sistemului se bazează pe arhitectura propusă de NIST. Am decis separarea indexării şi a modulelor de căutare, în detrimentul căutării directe în corpus a fiecărui termen al interogării, pentru a face căutarea mai rapidă, deoarece metoda de indexare simplifică căutarea [Buzo, 2013]. Astfel abordarea constă în două etape:

- Indexarea, adică recunoaşterea de fonem pe baza de date audio, de conţinut
- Căutarea, adică găsirea unui şir similar de foneme din conţinutul indexat, care se potriveşte cu cel al interogării, prin utilizarea unui algoritm de căutare DTW.

Recunoaşterea fonemelor este folosită pentru indexare, astfel tot conţinutul vorbit este transformat în şiruri de foneme. Pentru limbile slab-dotate, unde resursele sunt limitate, recunoaşterea de foneme este o alegere bună, deoarece nu avem suficiente date pentru a construi modele complexe de limbă de tip n-gram [Caranica, 2015].

Pentru experimente, am început cu două tipuri de caracteristici, pentru a reprezenta parametric vorbirea: MFCC şi PNCC, pentru o robusteţea îmbunătăţită, teoretic, la zgomot. În ceea ce priveşte modelul acustic (AM), în prima abordare, folosind decodoare de foneme bazate pe HMM, ne-am dorit să comparăm efectul utilizării de resurse multilingve faţă de cele monolingve şi pentru a realiza acest lucru, am construit şase modele acustice descrise în Tabelul 4. Am început cu modele acustice pentru fiecare limbă, apoi am folosit clasificarea IPA pentru a combina fonemele comune în modelele de limbă (LM) antrenate cu toate datele. Combinarea fonemelor comune este motivată de numărul mare de foneme obţinute în AM4, în cazul în care fonemele din diferite limbi sunt antrenate separat şi sunt privite ca entităţi diferite. Acest lucru ne-a permis să reducem numărul de foneme la 98, pentru modelul în mai multe limbi AM5. Am folosit un număr moderat de date de antrenare pentru limbile individuale, pentru a avea o antrenare echilibrată între limbi. Pentru comparaţie, am antrenat şi un modelul acustic suplimentar pentru limba română (AM6), cu o cantitate mare de date (64h) şi un set relativ mic de foneme (34).

ID	Language	LM no. phonemes	Training data [h]
AM1	Romanian	34	8.7
AM2	Albanian	36	4.1
AM3	English	75	3.9
AM4	Multilingual separate phones	145	16.7
AM5	Multilingual common phones (IPA)	98	16.7
AM6	RomanianBig	34	64

Tabelul 4 Seturile de date de antrenare folosite pentru abordarea HMM, pentru algoritmul de căutare

Suplimentar, am încercat introducerea unor noi limbi în faza de antrenare, pentru a vedea modul în care acestea se comportă, împreună cu o reţea neuronală bazată pe recunoaştere de foneme, de la BUT [BUT, 2015]. Acest decodor de foneme utilizează o divizare dependentă de context (STC) pentru extracţia de caracteristici, cu clasificatorii bazaţi pe reţele neuronale pentru a obţine posteriograme, în timp ce algoritmul Viterbi este folosi pentru decodare. Putem folosi ieşirea acestui instrument în algoritmul de căutare DTW ca şi caracteristici de intrare, pentru a face căutarea. Cu scopul de a utiliza limbi suplimentare pentru a construi parametrii caracteristici pentru recunoaştere fonemelor, am folosit sistemele pre-antrenate disponibile la [BUT, 2015], descrise în Tabelul 5.

ID	Limba	# foneme in LM	WER[%]
AM7	Czech	45	24.24
AM8	Hungarian	61	33.32
AM9	Russian	52	39.27

Tabelul 5 Sistemele antrenate folosind STC

Limbile utilizate pentru antrenarea sistemelor descrise în Tabelul 5 fac parte din baza de date SpeechDat-E a limbilor vorbite în Europa de Est [Speechdat-E, 2015]. Un alt motiv pentru a folosi aceste sisteme o reprezintă existenţa unor simboluri non-vorbire antrenate, care s-ar dovedi utile în medii acustice dificile, cum ar fi:

- „int" pentru zgomot intermitent
- „spk" pentru zgomotul vorbitorului (tuse etc)

- „pau” pentru zonele de liniște

Metoda de căutare propusă foloseşte un algoritm de Dynamic Time Warping (DTW) pentru a alinia un şir (o interogare) într-un anumit conținut. Căutarea nu se face pe întregul conținut, ci numai pe o parte a acestuia prin intermediul unei ferestre glisante proporțională cu lungimea interogării, unde atât interogarea cât şi conținutul în care se caută sunt un şir de foneme. Termenul este considerat detectat dacă scorul DTW trece de un prag. În plus, față de algoritmul clasic DTW, am inclus, în formula distanței, efectul lungimii interogării şi a împrăştierii detecției DTW. Efectul lor este ponderat, cu scopul de a găsi o configurație optimă. Precizia sistemului RVC utilizat pentru indexare joacă un rol important, deoarece algoritmul de căutare trebuie să compenseze pentru rata destul de mare de eroare a fonemelor (*PhER*).

Deoarece lungimea conținutului este de obicei mai mare decât lungimea interogării, comparația se face cu ajutorul unei ferestre glisante a cărei lungime este proporțională cu lungimea interogării. Pentru fiecare fereastră, alinierea este dată de scorul *s*:

$$s = (1 - PhER) \tag{1}$$

unde *s* este un scor de similitudine. Detecția se bazează pe un prag, care se determină empiric.

Metoda de detecție este îmbunătățită prin introducerea unei penalizări pentru interogările scurte şi împrăştierea detecției DTW. Penalizările sunt motivate de presupunerea că, pentru două interogări de diferite lungimi care se potrivesc cu conținutul lor la aceeaşi rată de eroare a fonemelor (*PhER*), potrivirea interogării mai lungi este mai probabil să fie cea corectă. Formula pentru scorul *s* este acum dată de ecuația [Buzo, 2013]:

$$s = (1 - PhER)(1 + \alpha \frac{L_Q - L_{Qm}}{L_{QM} - L_{Qm}})(1 + \beta \frac{L_w - L_S}{L_Q}) \tag{2}$$

unde L_Q este lungimea interogării, L_{QM} şi L_{Qm} sunt valorile maxime şi minime pentru L_Q, L_W este mărimea ferestrei, L_S este împrăştierea detecției DTW, iar α şi β sunt coeficienții care controlează nivelul penalizărilor introduse. Penalizările introduse în ecuația de mai sus sunt motivate de presupunerea că, pentru două interogări de diferite lungimi care se potrivesc cu conținutul lor la aceeaşi rată de eroare a fonemelor (*PhER*), potrivirea interogării mai lungi este mai probabil să fie cea corectă. În mod similar, detecțiile mai compacte DTW se presupun a fi mai probabile decât cele mai lungi.

Bibliografie

[BUT, 2015] BUT Phoneme recognizer based on long temporal context, Brno University of Technology, http://speech.fit.vutbr.cz/software/phoneme-recognizer-based-long-temporal-context, accessed august 2015.

[Buzo, 2013] Buzo A., Cucu H., Safta M., Burileanu C., “Multilingual Query by Example Spoken Term Detection for Under-Resourced Languages”, in Speech Technology and Human - Computer Dialogue (SpeD), (2013).

[Caranica, 2015] Alexandru Caranica, “Optimizations in spoken language recognition“, PhD Thesis, University “Politehnica” of Bucharest, Nov 2015 (scientific coordinator: prof. Corneliu Burileanu).

3.9 Activitatea 4.9. Implementarea infrastructurii şi algoritmilor de detecție multilingva a termenilor vorbiți

Sistemele de detecție a recurenței termenilor vorbiți (Spoken Term Discovery) identifica fragmente de vorbire repetitive din discursul ce conține vorbire, fără nici o informație despre limba acestuia [Park, 2008], pentru a construi clase de fragmente de vorbire similare. Abordările actuale pentru detecția recurenței termenilor vorbiți se bazează pe variante de tip Dynamic Time Warping (DTW), pentru a efectua în mod eficient o căutare într-un corpus de vorbire, cu scopul de a descoperi repetiții (clase similare) în vorbire (numite în continuare "termeni" sau "motive").

Metodologia propusă foloseşte un algoritm open-source de detecție a recurențelor audio, denumit MODIS [Catanese, 2013], bazat pe sistemul propus în [Muscariello, 2012]. Funcționarea sistemului urmează aşa numitul principiu descoperire a „fragmentelor”, adică de a căuta un segment audio mai scurt într-un segment mai mare, căutarea fiind realizată prin intermediul unei „variante de avansare” prin segmente de tip „dynamic time warping” (DTW). În acest sistem segmentul mai scurt este numit „seed”, în timp ce fragmentul mai mare e denumit „buffer”.

Algoritmul inspectează secvențele acustice prezente în buffer pentru a evalua dacă acesta conține orice repetiție a fragmentelor, apoi o decizie de asemănare este luată prin comparația scorului DTW cu un prag similitudine DTW a unei librării de motive. Dacă nu s-au găsit asemănări între fragmente, procesul „sare” la următorul fragment din buffer, pentru a căuta asemănări cu librăriile de motive. Dacă o asemănare este găsită, atunci librăria de motive este actualizată sau „învățată”, în final obținându-se „clustere” de vorbire similare. Algoritmul continuă până la finalizarea căutării tuturor fragmentelor în librăriile disponibile, prin repetiția proceselor de mai sus. După cum am menționat anterior, modelul de cluster corespunzător unei librării de motive este actualizat continuu. După ce întregul corpus a fost analizat, motivele găsite sunt comparate între ele, în termeni de suprapunere, iar elemente suprapuse sunt unite într-o singură

librărie. Acest proces este ilustrat în Figura 3. MODIS a fost proiectat pentru a lucra cu mai multe tipuri de caracteristici şi reprezentări parametrice a vorbirii. Astfel sunt implementate două tipuri de distanţe: distanţa euclidiană, în general utilizată împreună cu reprezentările spectrale, iar distanţa definită în ecuaţia de mai jos, pentru reprezentările cu probabilităţi posterioare:

$$d(a,b) = -\log\left(\sum_{i=0}^{N}(a_i \cdot b_i)\right) \quad (3)$$

Termenii implicaţi în ecuaţia de mai sus reprezintă cei doi vectori caracteristici pentru care se calculează distanţa (*a* şi *b*) şi lungimea lor (*N*).

Pentru experimentele de căutare a recurenţelor termenilor vorbiţi, am variat pragul de similitudine, păstrând în acelaşi timp restul parametrilor constanţi. Durata fragmentului a fost stabilită la 0.25 secunde, iar dimensiunea minimă a termenilor returnaţi e de 0.5 secunde, pentru a putea găsi cuvinte întregi. Lungimea bufferului e de 600 secunde, deoarece, în general, fişierele audio din bazele noastre de date pot avea mai puţine repetiţii ale aceluiaşi cuvânt. Modelul ales pentru a reprezenta clusterele termenilor a fost modelul median, în timp ce o matrice de verificare similitudini [Muscariello, 2011] a fost folosită pentru a potrivi modelul din cluster, de fragmentele analizate. În ceea ce priveşte distanţele utilizate, caracteristicile posteriorgram folosesc distanţa prezentată în ecuaţia anterioară (3), în timp ce MFCC şi vectorii de foneme utilizează distanţa euclidiană.

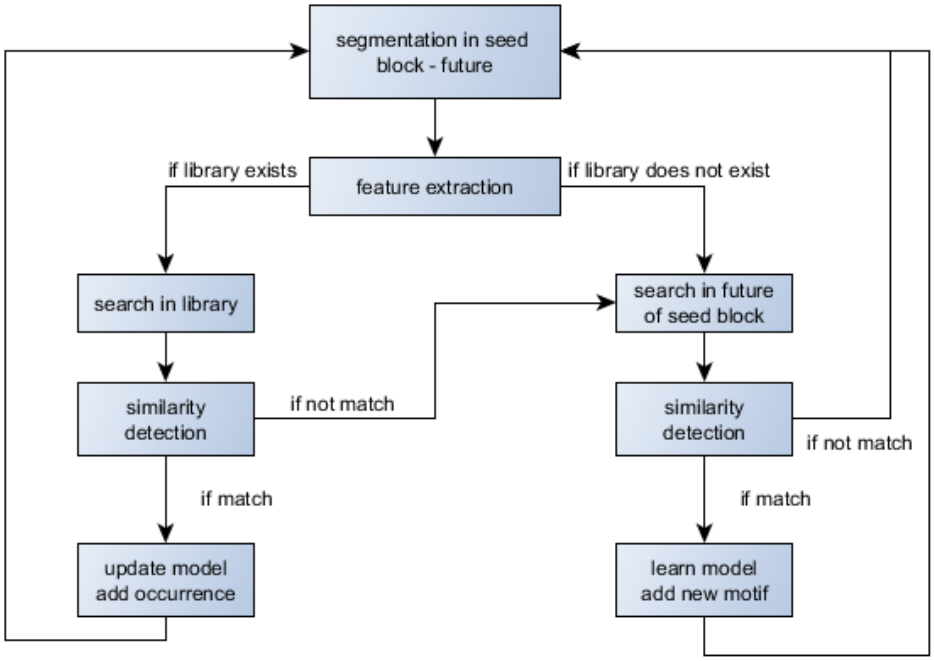

Figura 3 Arhitectura algoritmului MODIS, propus de [Catanese, 2013]

Pentru efectuarea sarcinii de detecţie a cuvintelor cheie într-o secvenţă de vorbire, a fost creat un serviciu web ce poate fi accesat online la adresa http://keywords.speed.pub.ro. Acesta este parte componentă a unui ansamblu de servicii, a cărui arhitectură poate fi observată în Figura 4 Arhitectură SpeeD Web Services.

Din punct de vedere funcţional, această aplicaţie este compusă din următoarele componente:

I. Interfaţă grafică (frontend) - permite accesul utilizatorului la mai multe facilităţi:
- crearea şi accesarea unui cont de utilizator, precum şi editarea datelor personale;
- încărcare a fişierelor audio sau text în contul de utilizator;
- crearea unor cereri de transcriere a fişierelor audio existente în contul de utilizator;
- crearea unor cereri de căutare a cuvintelor cheie ce se pot găsi în fişierele audio existente în contul de utilizator;
- crearea unor cereri de restaurare a diacriticelor în fişierele text existente în contul de utilizator.

II. Componente de management
- Baza de date: stochează informaţii despre utilizatori, cât şi date necesare efectuării proceselor descrise sau date rezultate în urma acestora;

- Manager de cozi: stochează cererile trimise de utilizatori prin API Gateway, precum şi informații despre starea acestor cereri (efectuate, în curs de procesare).

III. Componente de procesare a datelor

- Diarizator: etichetează datele audio, oferind informații temporale asupra segmentelor rostite de către un vorbitor. De asemenea, oferă informații despre natura segmentelor audio (vorbire sau non-vorbire);
- Speech Transcriber: realizează transcrierea segmentelor audio, oferind textul aferent acestora. Este bazat pe sistemul de recunoaştere automată a vorbirii antrenat cu utilitarul Kaldi.
- Transcription Postprocessor: realizează procesarea textelor obținute în urma transcrierii. Acest proces include formatarea paragrafelor, inserarea semnelor de punctuație, inserarea majusculelor;
- Diacritics Restorer: realizează restaurarea diacriticelor în fişiere text;
- Keyword Spotter: realizează detecția unor cuvinte cheie introduse de utilizator în textul aferent transcrierii fişierelor audio.

Pentru a efectua o operație de detecție a cuvintelor cheie, utilizatorul va iniția o cerere în care va specifica fişierul audio în care se va face căutarea, precum şi lista de cuvinte cheie. Cererea va fi introdusă in sistemul de cozi, iar de aici va fi preluată de către Diarizator, Speech Transcriber şi Keyword Spotter.

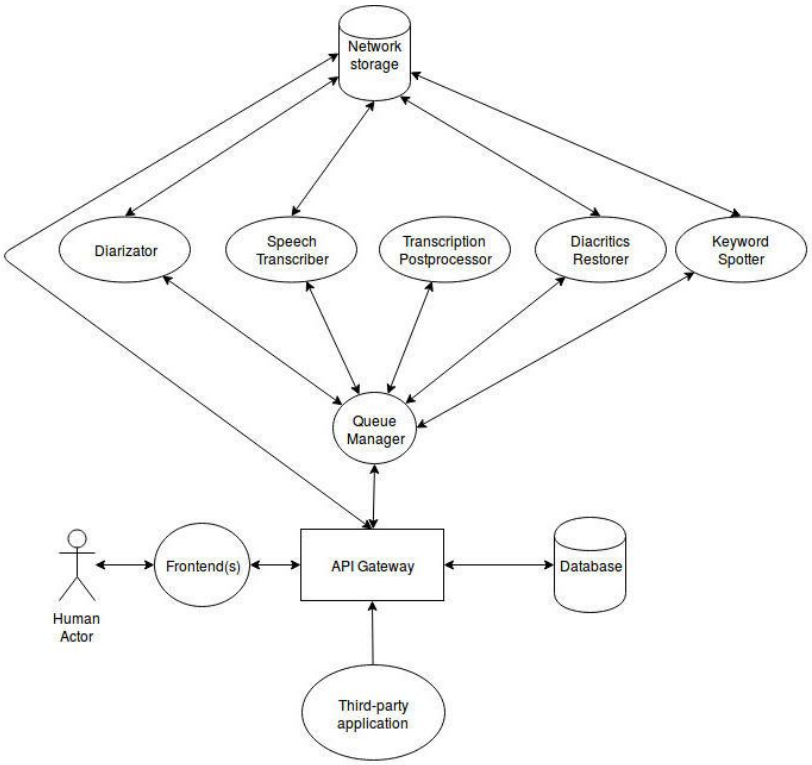

Figura 4 Arhitectură SpeeD Web Services

Pentru a efectua o operație de detecție a cuvintelor cheie, utilizatorul trebuie mai întâi să încarce un fişier audio în aplicație. Astfel, va fi efectuată o cerere de tipul HTTP POST către API Gateway pe terminalul /audio, care va conține tokenul obținut în urma autentificării, precum şi octeții audio în format Base64. API Gateway va stoca fişierul audio în Network Storage, iar informații despre acesta vor fi înregistrate în baza de date, fiind asociat fişierului şi un ID unic.

Mai departe, utilizatorul va iniția o nouă cerere de tip HTTP POST către API Gateway pe terminalul /kws, care va conține ID-ul fişierului audio în care se va face căutarea cuvintelor cheie, domeniul căruia aparține vorbirea din fişier (general, economic, politc, sportiv, etc.), precum şi lista cuvintelor cheie ce se doresc a fi detectate. ID-ul fişierului, domeniul de vorbire, tipul sarcinii (detecția cuvintelor cheie), precum şi statusul sarcinii, vor fi inserate în baza de date, în tabelul AudioJobs. În tabelul Keywords vor fi inserate cuvintele cheie.

API Gateway va pune o cerere de diarizare ce conține ID-ul fişierului audio în coada corespunzătoare componentei Diarizator. Acesta va prelua fişierul audio din Network Storage, va realiza operația de diarizare şi va pune informațiile rezultate în coada corespunzătoare componentei API Gateway. Componenta va prelua informațiile din coadă şi va actualiza tabelul AudioSegments din baza de date, completând cu informații despre fiecare segment (etichete temporale, genul vorbitorului, etc.). Apoi, vor fi puse în coada corespunzătoare componentei Speech Transcriber, N cereri de transcriere, asociate celor N segmente. Cererile conțin etichetele temporale ale fiecărui segment, precum şi domeniul de vorbire, folosit ulterior pentru selecția modelului acustic potrivit. Componenta Speech

Transcriber va prelua fişierul audio din Network Storage, va transcrie fiecare segment, iar la final va pune transcrierea rezultată în coada corespunzătoare componentei API Gateway. Acesta va prelua informaţiile din coadă şi va actualiza tabelul Transcriptions din baza de date. Totodata, va pune în coada corespunzătoare componentei Keyword Spotter, N cereri de detecţie a cuvintelor cheie. Cererile conţin transcrierile segmentelor şi ID-urile cuvintelor cheie. Keyword Spotter va prelua aceste informaţii din coadă şi va căuta fiecare cuvânt cheie în transcrierile furnizate. La final, va pune etichetele temporale între care se găsesc cuvintele cheie detectate, în coada corepunzătoare componentei API Gateway. Aceasta va citi informaţiile din coadă, va actualiza tabelul KeywordPositions, va actualiza tabelul AudioJobs, modificând statusul sarcinii de detecţie a cuvintelor cheie, acesta fiind acum terminat. De specificat însă faptul că la început, se va verifica dacă au fost efectuate în trecut operaţiile de diarizare sau transcriere pentru fişierul audio în cauză. În această situaţie, se va sări peste paşii respectivi.

Pentru a obţine rezultatul final, utilizatorul va iniţia o cerere de tip HTTP GET către API Gateway pe terminalul /kws/{job_id}, primind în răspuns o listă a cuvintelor cheie detectate, împreună cu momentele de timp la care acestea apar în fişierul audio.

Bibliografie

[Catanese, 2013] L. Catanese, N. Souviraà-Labastie, B. Qu, S. Campion, G. Gravier, E. Vincent, and F. Bimbot, "MODIS: an audio motif discovery software," in Proc. of INTERSPEECH, 2013.

[Park, 2008] A. Park and R. Glass, "Unsupervised pattern discovery in speech," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 1, pp. 186–197, 2008.

[Muscariello, 2012] A. Muscariello, G. Gravier, and F. Bimbot, "Unsupervised motif acquisition in speech via seeded discovery and template matching combination," IEEE Transactions on Audio, Speech, and Language Processing, vol. 20, no. 7, pp. 2031–2044, 2012.

3.10 Activitatea 4.10. Compararea si evaluarea performantelor tehnicilor implementate

Această activitate prezintă rezultatele decodării bazei de date, achiziţionate în limba Engleză (homeAuto4), în cadrul grupului de lucru, folosind modelele acustice antrenate pe corpusul TED-LIUM, în cele două utilitare integrate de către grupul SpeeD: CMU Sphinx şi Kaldi. Se compară modelele folosind metrica standard WER (Word Error Rate).

Sistem	Configuraţie	WER [%]
Kaldi	Tri 1	55
Kaldi	Tri 2 (LDA+MLLT)	54
Kaldi	Tri 3 (LDA+MLLT+SAT)	18
Kaldi	Tri3 (MMI)	17
Kaldi	NNET2	16
Sphinx	GMM-LM	49
Sphinx	GMM-FSG	49

Tabelul 6 Evaluarea performanţelor sistemului în limba Engleză

Se poate observa, cu uşurinţă, că utilitarul Kaldi oferă performanţe mult mai bune, cel puţin în limba engleză, datorită următorilor factori:

- Antrenarea pe baza de reţele neuronale (Kaldi), ceea ce a oferit un spor de performanţă pentru baza de date achiziţionată în limba engleză, după cum se observă. Avantajul major este reprezentat de existenta algoritmilor de antrenare ce folosesc reţele neuronale adânci (DNN), ce pot capta mai bine variaţiile frazelor rostite de vorbitorii non-nativi de limba engleză.
- Kaldi realizează şi o adaptare la vorbitor („speaker adaptation"), pentru a capta variaţiile din cadrul vorbirii, cum ar fi accentele non-native, stilul discursului, etc.
- CMU Sphinx nu poate astfel recupera diferenţele majore în vorbire, dintre baza de date de antrenare şi cea de decodare (pronunţie, stil, accentul limbii, etc).

Pentru această evaluare, s-a folosit o gramatică cu stări finite (FSG – Finite State Grammar), pentru ambele toolkit-uri, în cadrul sistemului prototip. Figura 5 prezintă o porţiune din acest model FSG, ce conţine, ca şi în limba română, un model de tip "garbage" pentru rejecţia propoziţiilor ce nu conţin comenzi.

```
grammar functii;

//-------------------------------------------------------------------------------------------------------------------------------------------------------
<multimedia> = <multimedia_on_off> | <multimedia_on_off_1> | <volume_settings> | <volume_change_1> | <radio_selection> | <source>;

<multimedia_on_off> = (start | stop | turn off | play) ([the] music |[the] radio |[the] tv);
<multimedia_on_off_1> = (play | run) ([the] music list | [the] music playlist | [the] playlist);

<volume_settings> = (set | change | make) (volume | sound) ([the] tv | music) to <decimal_number> percent;
<volume_change_1> = (decrease|increase|lower|make [the]) (volume | tv [quieter] | tv [volume]);

<radio_selection> = (i want to listen [to] | play) (radio europa fm | radio zu | radio pro fm);

<source> = (change|switch) (source | to hdmi | the tv | plug-in) [(source | source to hdmi | [source] to vga | source to plug-in)];

//-------------------------------------------------------------------------------------------------------------------------------------------------------
<exterior> = <sprinklers> | <front_gate> | <garage_door> | <front_door> | <modes>;

<sprinklers> = (start|stop|activate|deactivate|turn) ( [the] sprinklers | [the] irrigation | [the] irrigation system [on]);

<front_gate> = (lock|unlock|close|open) (the entrance gate | entrance);
<garage_door> = (lock|unlock|close|open) (the garage door | garage door | the garage);
<front_door> = (lock|unlock|close|open) ([the] front door | [the] entrance);

<modes> = (start|activate|stop|deactivate) (reading|read|dinner|movie) mode;
```

Figura 5 Gramatica cu stări finite, în limba Engleză

Pentru CMU Sphinx s-a realizat şi o etapă de "tuning" a parametrilor OOGP (Out-of-grammar Probability), RBW (Relative Beam Width) şi WIP (Word Insertion Probability), pentru gramatica de tip FSG.

3.11 Activitatea 4.11. Crearea unui prototip de sistem de recunoaştere multilingvă a vorbirii

Prototipul sistemului de recunoaştere multilingvă a vorbirii constă într-un kit de dezvoltare Raspberry PI 3, unde cu ajutorul utilitarului Sphinx, a fost implementat un sistem mixt, ce realizează atât detecţia cuvintelor cheie (DCC), cât şi recunoaşterea automată a vorbirii (RAV).

Odată cu alimentarea kitului, este pornit un proces de DCC ce preia date audio de la microfon şi încearcă să stabilească dacă printre acestea apare cuvântul cheie „Casandra". Atunci când cuvântul cheie a fost detectat, va începe un proces de RAV ce utilizează o gramatică cu stări finite, care va stabili tipul comenzii rostite de utilizator. Comenzile sunt cele definite în etapele anterioare ale proiectului.

Integrarea modulului de recunoaştere a vorbirii cu controlerul de voce se realizează prin trimiterea către acesta din urmă a unei cereri de tip HTTP POST, ce conţine transcrierea comenzii. Controlerul de voce înţelege comanda primită şi o efectuează fizic (aprinde lumina, modifică temperatura, etc.), iar apoi întoarce în răspunsul HTTP un text ce informează asupra efectuării cu succes a comenzii.

Integrarea modulului de recunoaştere a vorbirii cu sistemul de sinteză a vorbirii se realizează prin trimiterea către acesta din urmă a unei cereri de tip HTTP GET, ce conţine textul întors de către controlerul de voce. Acest text va fi sintetizat, obţinându-se un fişier audio care va fi întors în răspunsul HTTP.

În final, sistemul implementat pe Raspberry PI redă la difuzor fişierul audio sintetizat, informând utilizatorul despre efectuarea comenzii. Apoi, face comutarea de la modul de funcţionare RAV la modul de funcţionare DCC. Astfel, sistemul aşteaptă ca utilizatorul să rostească din nou cuvântul cheie „Casandra" urmat de o nouă comandă vocal; tot procesul descris mai sus se reia. Acest proces este ilustrat în Figura 6 Organigrama sistemului DCC şi RAV.

Pentru implementarea procesării multilingve, au fost propuse două variante. Varianta 1 presupune:

- crearea şi utilizarea mai multor modele (acustice, fonetice si lingvistice) distincte pentru fiecare limbă în parte
- existenţa unui mecanism de selecţie statică a limbii de interacţiune cu sistemul
- comutarea manuală pe limba dorită şi apoi interacţiunea cu sistemul în acea limbă

Concret, această variantă implică conectarea la kitul Raspberry PI a unui buton cu două stări, ce are rolul de a selecta limba dorită (română sau engleză). În funcţie de poziţia butonului, modulul de recunoaştere automată a vorbirii va porni având încărcate modelele corespunzătoare uneia dintre limbi. Ulterior, dacă se doreşte interacţiunea cu sistemul în cealaltă limbă, este necesară comutarea manuală a butonului de selecţie.

Varianta 2 presupune crearea şi utilizarea antrenarea unei singur set de modele (acustice, fonetice şi lingvistice) folosind o bază de date de vorbire multilingvă. Astfel, fonemele sunt antrenate la comun, a a cum a fost descris în Activitatea 4.7. Dezvoltarea unor modele acustice multilingve si pentru limba engleza (partea II) Modelul de limbă de tip gramatică cu stări finite se obţine prin concatenarea regulilor corespunzătoare gramaticilor fiecărei limbi.

Varianta 1 are avantajul de a fi mai simplu de gestionat la momentul dezvoltării. Practic, pentru fiecare limbă în parte se crează modele acustice, fonetice şi lingvistice distincte, utilizând baze de date de vorbire monolingve şi cunoştinţe lingvistice specifice limbii în cauză. La limită, în cazul în care sunt multe limbi, dezvoltarea se poate realiza de echipe separate (fiecare fiind responsabilă de o singură limbă). Evaluarea acestor sisteme se face de asemenea mult

mai simplu (cu baze de date de vorbire monolingve). Un alt avantaj major al acestei variante este acela că acuratețea sistemului nu depinde de numărul de limbi suportate şi este fără îndoială mai mare decât în cazul variantei 2. Dezavantajul principal al acestei variante este reprezentat de mecanismul manual de selecție a limbii, mecanism ce ar putea părea rudimentar într-un sistem inteligent precum cel prezentat mai sus. Cu toate acestea, în contextul curent (casă inteligentă cu care se poate itneracționa prin voce) este puțin probabil ca schimbarea limbii de interacțiune să fie realizată prea frecvent. Cazul real de funcționare este acela în care limba este configurată în momentul instalării sistemului (dintr-o listă de limbi posibile), iar utilizatorul interacționează întotdeauna cu sistemul folosind acea limbă.

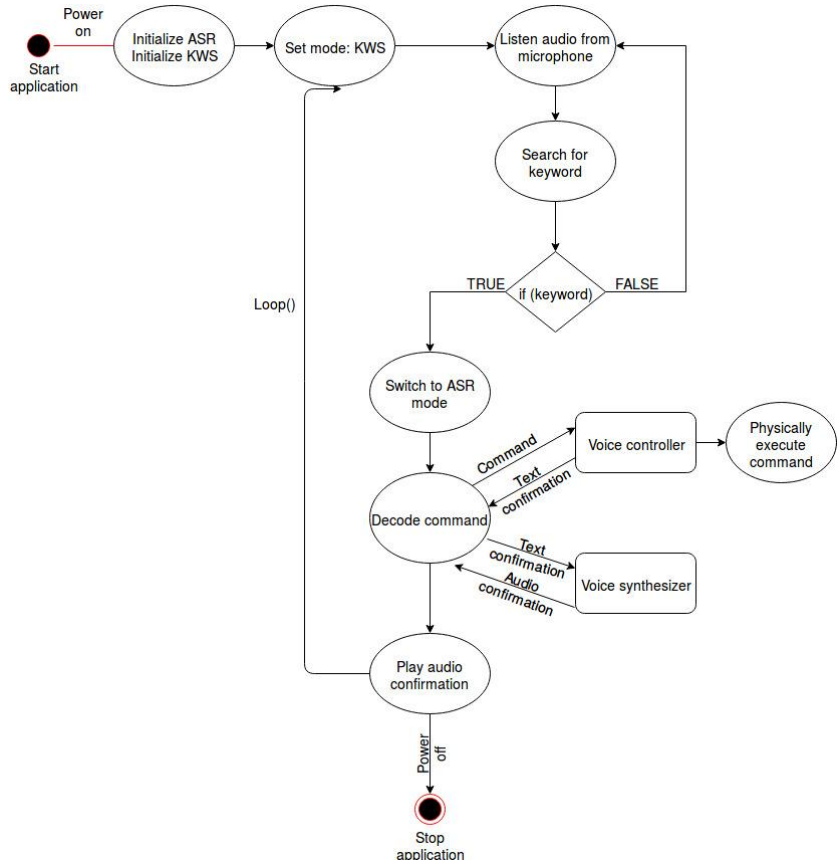

Figura 6 Organigrama sistemului DCC şi RAV

Varianta 2 are avantajul de a fi mai scalabilă şi mai simplu de gestionat la momentul utilizării pentru că întotdeauna va exista un singur model acustic, un singur model fonetic şi un singur model lingvistic (indiferent câte limbi va permite sistemul). Dezavantajul major al acestei variante este acela că acuratețea sistemului (acuratețea de recunoaştere a comenzilor) este mai mare. Motivul este simplu de înțeles: cu cât sunt mai multe comenzi posibile, în mai multe limbi posibile, cu atât probabilitatea de confuzie a unei comenzi cu o alta creşte.

În prototipul realizat a fost implementată varianta 1 de procesare multilingvă a vorbirii.

Întregul sistem de recunoaştere multilingvă a vorbirii a fost implementat şi într-o variantă demonstrativă, fizic funcțională (”demo bench”), ce a constat în configurarea şi integrarea următoarelor dispozitive şi periferice:

- Kit Philips Hue, ce conține 3 becuri LED de 10W A19 şi E27, controlabil prin intermediul unui API de tip REST (GET/POST)
- Sursă KNX + Router IP KNX, pentru controlul unor sisteme de automatizare clădiri, cum ar fi un termostat
- Termostat KNX cu afişaj digital, pentru reglarea automată a temperaturii
- Intel NUC N3700, cu 8GB ram şi SSD de 120GB, pentru partea centrală (server) a aplicației demonstrative
- Router WiFi Mikrotik RB951Ui-2HnD, cu suport USB, ce permite conectarea la modem-uri 3G / 4G, pentru a oferi conectivitate IP sistemului, independent de locație
- Două Raspberry Pi 3 Model B, ce rulează aplicația de recunoaştere a cuvintelor cheie ”Casandra” şi a comenzilor, ce le trimite prin intermediul unor variabile POST către controller
- Boxe pentru Raspberry Pi, cu scopul de a reda comenzile sintetizate
- Două microfoane USB, pentru captarea sunetului

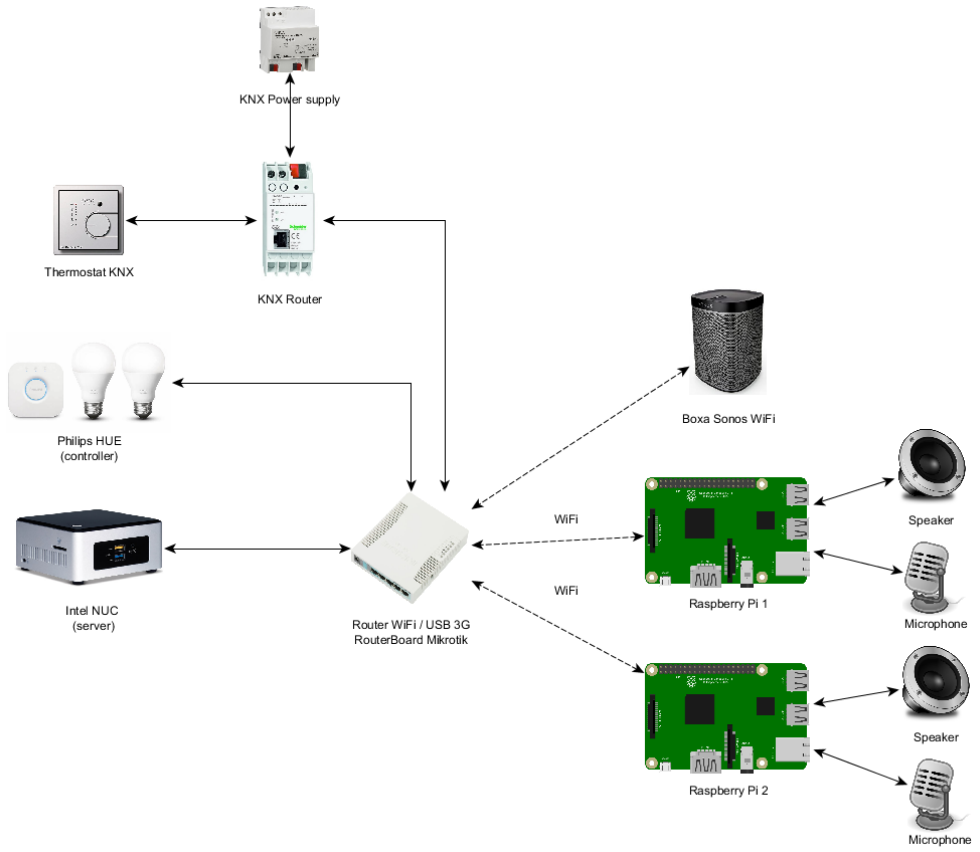

Figura 7 Arhitectura proptotipului hardware, demonstrativ

Scopul final al acestui prototip hardware este de a demonstra, într-un spațiu cât mai compact şi cu un consum de energie cât mai mic, un sistem inteligent de control al unei încăperi, prin intermediul vorbirii, în două limbi (română + engleză). Figura 7 prezintă arhitectura prototipului hardware şi modul de conectare al dispozitivelor şi perifericelor.

Partea centrală (serverul) o reprezintă mini PC-ul Intel NUC, pe care rulează proxy-ul de voce (ComfortClik), sub sistem de operare Windows. Sintetizatorul de vorbire (modulul TTS) şi controllerul de voce (modulul NLP) sunt integrate într-o maşină virtuală Linux (Ubuntu Server 16.04 LTS), virtualizată cu ajutorul proiectului open-source Oracle VirtualBox. Tot în această maşină virtuală se află şi modelele acustice, de limbă, pentru sintetizare, etc. Router-ul WiFi Mikrotik, pe lângă asigurarea conectivității IP la internet, oferă şi partea de switch şi routare pentru toate dispozitivele TCP/IP din Figura 7. Dispozitivele Raspberry Pi realizează detecția de cuvinte cheie şi recunoaşterea comenzilor rostite de utilizator şi comunică, prin intermediul REST, cu controllerul de voce şi sintetizatorul de vorbire. Din punct de vedere hardware la dispozitivele Raspberry Pi sunt conectate microfoane pentru preluarea comenzilor şi difuzoare pentru redarea răspunsurilor furnizate de casa inteligentă. Proxy-ul de voce (ComfortClick) asigură controlul, tot prin comenzi de tip REST, a becurilor inteligente Philips HUE, iar router-ul IP KNX controlează temperatura ambientală, prin intermediul termostatului, tot KNX.

Astfel, acest sistem prototip simulează o întreagă arhitectură ”inteligentă”, controlată prin voce şi vorbire naturală.s

3.12 Activitatea 4.12. Crearea unui sistem de sinteza a vorbirii pentru limba romana

Sistemul de sinteză a vorbirii dezvoltat în cadrul proiectului ANVSIB este unul bazat pe corpus, având la bază (a) un sistem de analiză a textului, responsabil pentru extragerea de informații lingvistice relevante pentru sinteză vorbirii (explicații în cele ce urmează) şi (b) un sistem de sinteză a vorbirii pornind de la informațiile lingvistice obținute de sistemul de analiză a textului. Extragerea de informații lingvistice relevante pentru sinteza vorbirii este un proces complex, prin care se urmăreşte conversia unei propoziții/paragraf într-o secvență de atribute discrete ce serveşte procesului de selectare/generare de unități acustice necesare în procesul de generare de voce. Pornind de la sistemul de analiză a textului deja existent, în cadrul acestei activități, RACAI a revizuit întreg lanțul de procesare şi a introdus un set nou de unelte de prelucrare a limbajului natural având ca scop îmbunătățirea modelelor deja existente, micşorarea dimensiunii modelelor şi reducerea complexității de calcul. Astfel, noul set de instrumente foloseşte un perceptron cu decodor Viterbi pentru etichetare morfosintactică şi o cascadă de clasificatori bazați pe arbori de decizie pentru

stabilirea atributelor morfolexicale. Întreg lanțul de prelucrare a fost testat pe mai mult de 50 de limbi în cadrul competiției CONLL Shared Task on Universal Dependencies Parsing, rezultatele fiind disponibile online[1].

De asemenea, RACAI a lucrat la îmbunătățirea şi extinderea sintetizatorului vocal, mai multe detalii fiind oferite în cadrul secțiunilor (3.13, 3.14 şi 3.15).

3.13 Activitatea 4.13. Crearea unui sistem de sinteza a vorbirii pentru limba engleza

Modelul de analiză a textului pentru limba engleză are la bază modelele antrenate în cadrul competiției CONLL, având în plus un sistem de transcriere fonetică pentru limba engleză ce a fost antrenat folosind lexiconul CMUDict. Modelul acustic folosit în experimentele noastre a fost pus la dispoziție în cadrul Blizzard Challenge 2016, modelele acustice putând fi folosite în condițiile impuse de organizatori.

3.14 Activitatea 4.14. Cercetarea si implementarea unui nou modul de prozodie

Generarea elementelor prozodice este în continuare un proces extrem de complicat în sinteza vorbirii pornind de la text. Principalul impediment constă în faptul că, pe lângă relația de suprafață dintre text şi elementele prozodice, vorbitorii umani folosesc prozodia atât pentru a transmite propriile stări emoționale către interlocutori cât şi pentru a facilitata înțelegerea mesajelor complexe prin scoaterea în evidență a unor cuvinte cu încărcătură cognitivă sau a acelor cuvinte care ar putea fi înțelese greşit de către interlocutori din motive subiective sau obiective (de exemplu cuvinte care se pronunță asemănător sau cuvinte rare ce nu sunt utilizate în vorbirea de zi cu zi).

Astfel, principala strategie în sinteza prozodică, în lipsa unor reguli clare şi bine studiate de către foneticieni, este folosirea unui set exhaustiv de atribute extrase din text, ce pot contribui la construirea unei linii melodice corelată cu o cadență variabilă în rostirea cuvintelor ce ține în special de stilul particular al vorbitorului folosit în crearea corpusului. Conform rezultatelor raportate în cercetare precum şi a propriei experiențe în realizarea de sisteme de sinteză a vorbirii, setul de trăsături folosit de noi este:

(a) Contextul fonetic: fonemul curent, însoți de două foneme aflate la dreapta şi două foneme aflate la stânga (însoțite de atribute specifice: consoană/vocală, mod de articulare etc.), precum şi poziția fonemului în silabă (început, final, mijloc)
(b) Silaba curentă: însăşi silaba, în cazul în care aceasta este frecventă (mai mult de 5[2] apariții în corpus), accent lexical (da/nu), a câta silaba din cuvânt, a câta silabă din propoziție, numărul de silabe din cuvânt, numărul de silabe din propoziție, numărul de silabe de la ultimul semn de punctuație până la poziția curentă, numărul de silabe până la următorul semn de punctuație
(c) Următorul semn de punctuație/tipul propoziției (declarativă, exclamativă, interogativă)
(d) Informații morfosintactice: poziția cuvântului curent în grupuri sintactice (Ion, 2007) (început, mijloc, final), partea de vorbire a cuvântului curent.

Stabilirea modului în care aceste atribute influențează prozodia se face în mod automat folosind arbori de decizie construiți pe baza principiului Minimum Description Length (MDL). Astfel, sistemul construieşte 3 arbori de decizie, având ca scop modelarea spectrală, a frecvenței fundamentale a vocii precum şi a duratei de rostire. Analiza se face astfel:

(a) În prealabil se foloseşte un aliniator la nivel de fonem pentru a obține corespondența între text şi vocea înregistrată (în cazul nostru am utilizat HTK)
(b) Fiecare fonem este ulterior modelat folosind un Modele Markov cu 5 stări (doar trei emit)
(c) Parametrii spectrali (MLSA sau STRAIGHT), durata şi frecvența fundamentală sunt extrase din fişierele acustice
(d) Pe baza alinierilor se construiesc arborii de decizie, având ca noduri întrebări referitoare la atributele lingvistice (de exemplu „Fonemul curent este ‚A': DA/NU) şi în frunze media şi deviația standard calculate pe toate eşantioanele acustice care ajung în frunză în faza de antrenare
(e) În timpul rulării sistemul generează automat o secvență de durate/frecvență fundamentală şi spectru, prin parcurgerea arborilor folosind atribute lingvistice extrase din propoziția ce trebuie sintetizată

3.15 Activitatea 4.15. Integrarea cu HTS

În cadrul acestei activități RACAI a dezvoltat un modul de sinteză parametrică statistică ce suportă atât analiză de tip MLSA cât şi STRAIGHT. Modelarea spectrală, a duratei de rostire şi a frecvenței fundamentale a vocii este realizată în prezent folosind arborii de decizie descrişi anterior, menționând însă faptul că orice altă metodă poate şi integrată cu sistemul actual (în experimentele noastre am folosit o rețea bidirecțională de tip LSTM pentru generarea frecvenței fundamentale, model care însă nu a fost folosit în sistemul final, datorită complexității crescute de calcul).

---

[1] http://universaldependencies.org/conll17/results.html

[2] Valoare stabilită experimental

Sistemul este integral scris în limbajul de programare JAVA, fără dependinţe externe de funcţionare, cu menţiunea că, pentru generarea arborilor de decizie folosim sistemul HTS.

3.16 Activitatea 4.16. Optimizarea si configurarea sistemului de sinteza a vorbirii

În această etapă am experimentat cu un număr mare de atribute lingvistice şi praguri de includere a acestora în modelele prozodice, având ca scop obţinerea unor rezultate performante atât din punct de vedere al calităţii vocii (naturalețe şi inteligibilitate) cât şi din punct de vedere al cerinţelor computaţionale:

(a) modele noastre sunt reduse ca dimensiunea datorită utilizării modelelor parametrice statistice
(b) oferă rezultate stabile, fără degradări în calitatea vorbirii şi fără a introduce artefacte sonore neplăcute
(c) complexitate redusă: cele mai mare cerinţe de calul vin din partea etichetării morfosintactice ($O(k^2n)$) şi din procesul de generare a vocii folosind STRAIGHT (motiv pentru care sistemul poate rula şi pe baza filtrului MLSA)
(d) Pachetele utilizate în momentul rulării nu necesită dependinţe externe, putând fi folosite în orice mediu de producţie (inclusiv sisteme ARM)
(e) Cerinţele de memorie sunt extrem de mici, sistemul putând rula sinteză MLSA în doar 16MB RAM

Dimensiunea totală a modelelor este redată în Tabelul 7.

Tabelul 7 Dimensiunea modelelor de sinteză a vorbirii

Limbă	Dimensiune model PLN	Dimensiune model acustic
Engleză	257MB	7.5MB
Română	2.9MB	3.8MB

3.17 Sumar al realizărilor / rezultatelor

Nr.	Denumirea rezultatului	Denumire activitate	Procent de realizare
1.	*D4.1 Modele acustice multilingve si pentru limba engleza*	*Activitatea 4.7. Dezvoltarea unor modele acustice multilingve si pentru limba engleza (partea II)*	*100%*
2.	*D4.2 Model functional de detectie multilingva a termenilor vorbiti*	*Activitatea 4.9. Implementarea infrastructurii si algoritmilor de detectie multilingva a termenilor*	*100%*
3.	*D4.3 - Prototip de sistem de recunoastere multilingva a vorbirii*	*Activitatea 4.11. Crearea unui prototip de sistem de recunoastere multilingva a vorbirii*	*100%*
4.	*D4.4 - Model functional de sinteza a vorbirii pentru limba romana*	*Activitatea 4.12. Crearea unui sistem de sinteza a vorbirii pentru limba romana*	*100%*
5.	*D4.5 - Model functional de sinteza a vorbirii pentru limba engleza*	*Activitatea 4.12. Crearea unui sistem de sinteza a vorbirii pentru limba engleza*	*100%*
6.	*D4.6 - Prototip multilingv de sinteza a vorbirii*	*Activitatea 4.16. Optimizarea si configurarea sistemului de sinteza a vorbirii*	*100%*
7.	*D4.7 - Prototip de camera inteligenta controlabil prin voce*	*Activitatea 4.5. Evaluarea prototipului*	*100%*