# Roboţii şi Societatea: Sisteme Cognitive pentru Roboţi Personali şi Vehicule Autonome

**Număr contract** 72PCCDI **din** 01/03/2018

## Etapa III - 2020

# Raport ştiinţific şi tehnic

**Consorțiu**
Coordonator proiect: UNIVERSITATEA POLITEHNICA DIN BUCURESTI
P1: INSTITUTUL DE CERCETARI PENTRU INTELIGENTA ARTIFICIALA „MIHAI DRAGANESCU"
P2: INSTITUTUL DE MATEMATICA "SIMION STOILOW" AL ACADEMIEI ROMANE
P3: UNIVERSITATEA TEHNICA DIN CLUJ - NAPOCA
P4: UNIVERSITATEA "DUNAREA DE JOS"

**Site proiect** http://aimas.cs.pub.ro/robin/

**CUPRINS**

# 1. Introducere

ROBIN este un proiect centrat pe utilizator care proiectează sisteme şi servicii pentru utilizarea roboţilor într-o societate digitală interconectată și permite companiilor realizarea de produse și servicii complexe, inteligente și performante destinate utilizatorilor și societății în ansamblu. Proiectul se referă la o gamă diversă de roboţi: roboţi asistivi care vin în sprijinul persoanelor cu nevoi speciale, roboţi de interacţiune cu clienţii şi roboţi software care pot fi instalaţi pe vehicule în scopul realizării unei conduceri autonome sau semi-autonome. Proiectul combină tehnici și tehnologii avansate de inteligență artificială, interacțiune om-robot, interacțiune cu un mediu pervasiv, arhitecturi și prelucrări în Cloud, arhitecturi Cloud-Edge si Cloud-Robotics.

Figura 1.1 prezintă conceptul general al proiectului, atât funcțional cât și arhitectural, prin care am dezvoltat sisteme și servicii științifice și tehnologice performante, construite peste platforme interoperabile, cu capacități de structurare și procesare inteligentă a datelor și oferite comunității științifice și agenților economici.

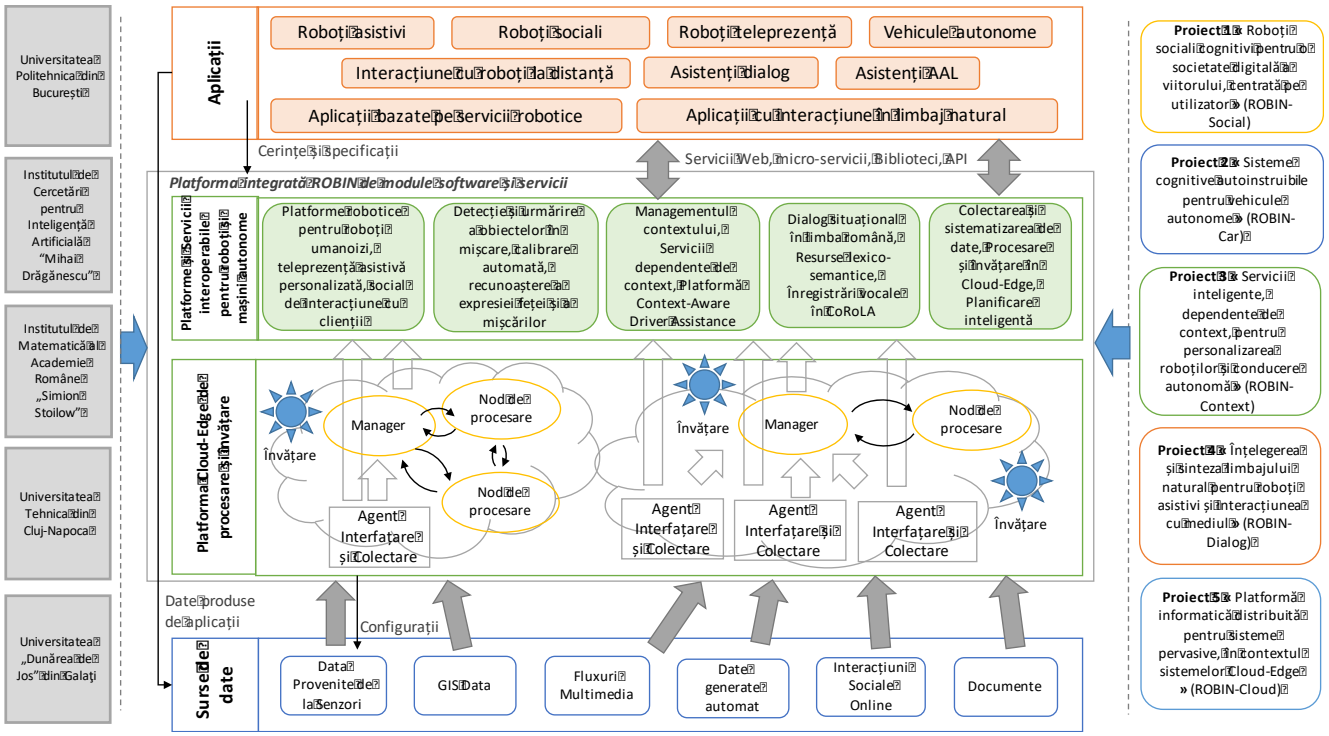


*Figura 1.1. Prezentarea conceptului integrat a proiectului ROBIN*

Proiectul ROBIN este alcătuit din cinci proiecte componente:

**ROBIN-Social**: ***Roboți sociali cognitivi pentru o societate digitală a viitorului, centrată pe utilizator.*** Scopul proiectului este realizarea unei soluții integrate și ușor configurabile pentru personalizarea roboților asistivi (pt. persoane cu nevoi speciale) și sociali (pt. clienți), soluție bazată pe tehnici avansate de inteligență artificială care pot oferi roboților un caracter cognitiv și autonom.

**ROBIN-Car**: ***Sisteme cognitive autoinstruibile pentru vehicule autonome.*** Scopul proiectului constă în dezvoltarea de metode de vedere computaţion- ală care să rezolve o gamă mai largă şi mai sofisticată de sarcini de asistenţă în pilotaj, realizarea unor module inteligente pentru „Hands-off driving" și „Automated driving" și un sistem prototip care va fi testat pe un autovehicul electric semi-autonom.

**ROBIN-Context**: ***Servicii inteligente, dependente de context, pentru personalizarea roboţilor şi conducere autonomă.*** Scopul proiectului este crearea unei platforme suport pentru definirea / reprezentarea semantică și gestiunea facilă și eficientă a datelor ce devin context în scenarii de asistență robotică personalizată și sisteme ADAS (Advanced Driving Assistance Systems).

**ROBIN-Dialog**: ***Înțelegerea și sinteza limbajului natural pentru roboți asistivi și interacțiunea cu mediul.*** Scopul proiectului presupune dezvoltarea unei serii de scenarii pentru câteva micro-lumi și tehnologia de prelucrare a limbii române pentru dialoguri situaționale în aceste micro-lumi.

**ROBIN-Cloud**: ***Platformă informatică distribuită pentru sisteme pervasive, în contextul sistemelor Cloud-Edge***. Scopul proiectului îl constituie crearea unei platforme suport pentru colectarea de date provenind de la senzorii unor sisteme suport pentru roboți (ex. IoT), oferirea de mecanisme de procesare și învățare combinând modele Cloud cu dispozitive aflate aproape de sursa de colectare, oferirea de biblioteci suport pentru procesare inteligentă / semantică a datelor, și în final dezvoltarea de servicii Web centrate spre agenți economici interesați de folosirea datelor stocate la nivelul platformei.

# 2. Proiectul component ROBIN-Social

**Parteneri ai proiectului ROBIN-Social**

UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI (CO)
INSTITUTUL DE CERCETĂRI PENTRU INTELIGENȚĂ ARTIFICIALĂ „MIHAI DRĂGĂNESCU"
INSTITUTUL DE MATEMATICĂ "SIMION STOILOW" AL ACADEMIEI ROMÂNE

## 2.1 Definitivarea platformei AMIRO

S-a realizat implementarea produsului robotic asistiv personalizat prin rafinarea implementării realizate în etapa II și particularizarea componentelor la situațiile specifice, de exemplu dialogul, recunoașterea persoanelor și asociere cu informațiile medicale, personalizarea mediului înconjurător. Implementările scenariilor au fost testate cu o singura persoana, cu persoane diferite, dar si cu un grup de persoane aflate in fata robotului. Rezultatele prezintă un comportament precis al robotului în funcție de situație.

S-a realizat definitivarea implementării, testării și evaluării platformei pentru roboți sociali AMIRO (AMbIent RObotics), implementată pe robotul Pepper, dar generalizabilă pentru orice robot compatibil cu sistemul de operare ROS2 (figura 2.1). Platforma dezvoltată funcționează pe o arhitectură care urmează tendințele recente ale roboticii bazate pe edge și cloud robotics. Platforma cuprinde un set de module funcționale care facilitează compoziția unui comportamentului complex: (i) navigarea și evitarea obstacolelor, (ii) recunoașterea persoanei și estimarea coordonatelor, (iii) recunoașterea activității umane, (iv) recunoașterea vorbirii, procesarea comenzilor și gestionarea dialogului, (v) integrarea cu medii inteligente și (vi) comportare de tip Belief-Desire-Intention pentru gestionarea comportării robotului. Platforma cuprinde o interfață care permite compunerea funcționalității modulelor individuale în comportamente complexe.

În timp ce multe lucrări din literatură se concentrează pe evaluarea sistemelor de robotică socială în ansamblu, puține dintre ele oferă un punct de referință pentru performanța fiecărei capacități individuale, utilizând fluxuri de date colectate direct de pe platforma robotului. În special, aceste observații sunt valabile cu atât mai mult în cazul robotului Pepper. O contribuție importantă este, prin urmare, aceea de a efectua teste cuantificabile ale fiecărui modul funcțional individual, pe baza seturilor de date colectate direct de la robotul Pepper în diverse setări de laborator. Experimentele documentate oferă o referință pentru robustețea platformei de robotică socială dezvoltată și informează despre avantajele și dezavantajele inerente hardware-ului Pepper sau perfectibile în implementarea noastră actuală.

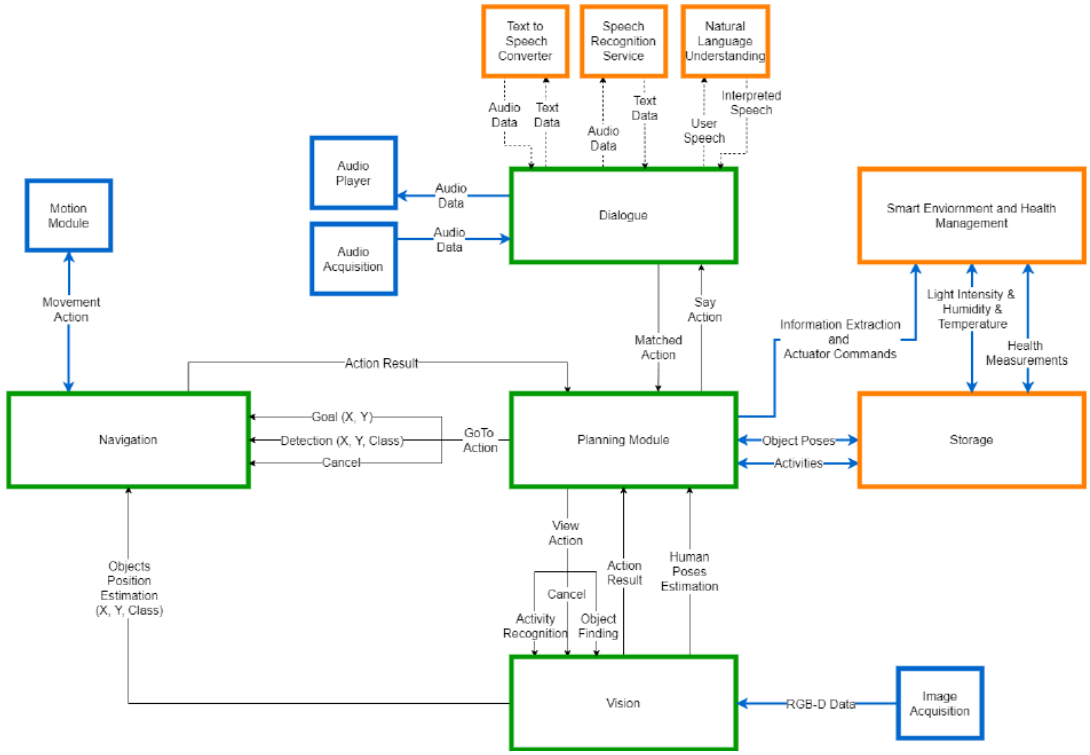


*Figura 2.1 Arhitectura sistemului AMIRO - diagramă bloc. Cu verde sunt marcate componentele cloud-edge, cu portocaliu sunt reprezentate servicii cloud și cu albastru modulele care funcționează pe robot. Săgețile negre reprezintă un apel intern, cele punctate reprezintă un apel la un serviciu cloud și cele albastre reprezintă o comunicare prin ROS.*

Am realizat evaluarea calitativă a platformei dezvoltate prin intermediul unui scenariu care evidențiază interacțiunea dintre toate modulele de funcționalitate disponibile. Scenariul implică primirea unei notificări de la un serviciu extern de gestionare a sănătății, căutare proactivă și navigare către utilizator, interacțiunea în limbaj natural și recunoașterea activității. De asemenea, am integrat platforma cu un framework de management al stării de sănătate a utilizatorului (dezvoltat anterior de colectivul de la UPB) care permite monitorizarea activității utilizatorului, a parametrilor de sănătate (ritm cardiac, oxigen în sânge, etc.) și a diferitelor dispozitive domotice. Platforma este disponibilă și poate fi controlată de pe diferite dispozitive, de exemplu desktop, tabletă sau telefon mobil.

## 2.2 Scenariu de validare și evaluare

Pentru a testa robustețea sistemului robotic propus, a fost realizat și implementat un scenariu de evaluare. Scenariul încorporează principalele componente ale platformei și ilustrează o parte a funcționalităților de bază ale framework-ului integrat, după cum urmează:

- recunoașterea și înțelegerea vocii;
- găsirea unei persoane folosind componenta vizuală;
- estimarea poziției unei ținte detectate;
- recunoașterea acțiunilor umane;
- explorarea și navigarea mediului;
- interacțiunea sistemului inteligent cu mediul;
- planificarea și execuția acțiunilor.

În scenariul propus, robotul interacționează cu o persoană pe care trebuie să o identifice și să o asiste la realizarea unei activități de bază, după cum urmează:

- Modulul Smart Environment and Health Management trimite o alertă robotului cu privire la temperatura din mediu și la modificarea parametrilor de sănătate.
- Robotul inițiază căutarea vizuală a persoanei în mediul în care este poziționat. Dacă persoana nu este identificată, scenariul se oprește.

- Dacă persoana a fost identificată, robotul merge la poziția în care a fost identificată aceasta. Dacă robotul nu poate ajunge în poziția în care a fost identificată persoana, scenariul se oprește.
- Dacă alerta pe care a primit-o robotul de la modulul Smart Environment and Health Management este legată de o temperatură ridicată, atunci robotul anunță notificarea de hidratare și așteaptă o confirmare audio de la utilizator.
- Dacă se primește confirmarea, modulul de Recunoaștere a acțiunilor este declanșat pentru a recunoaște activitatea bea apă. Dacă acțiunea nu este recunoscută, robotul va solicita o validare audio a activității efectuate.
- Dacă confirmarea nu este primită, robotul va întreba utilizatorul dacă acțiunea de hidratare trebuie amânată.
- Pepper trimite o confirmare către modulul Smart Environment and Health Management.

Figura 2.2 ilustrează o reprezentare vizuală a interacțiunii dintre sarcini incluse în scenariul propus. Schema de culori utilizată în figură prezintă componentele care se ocupă de realizarea sarcinilor: portocaliu - modulul Vision, albastru - modulul de Navigare, verde - modulul de Dialog, roșu - modulul de Planificare. Interacțiunea cu componenta Smart Environment and Health Management este mediată de componenta de Planificare.

Videoclipurile[1] demonstrează o implementare a scenariului propus în cadrul laboratorului.

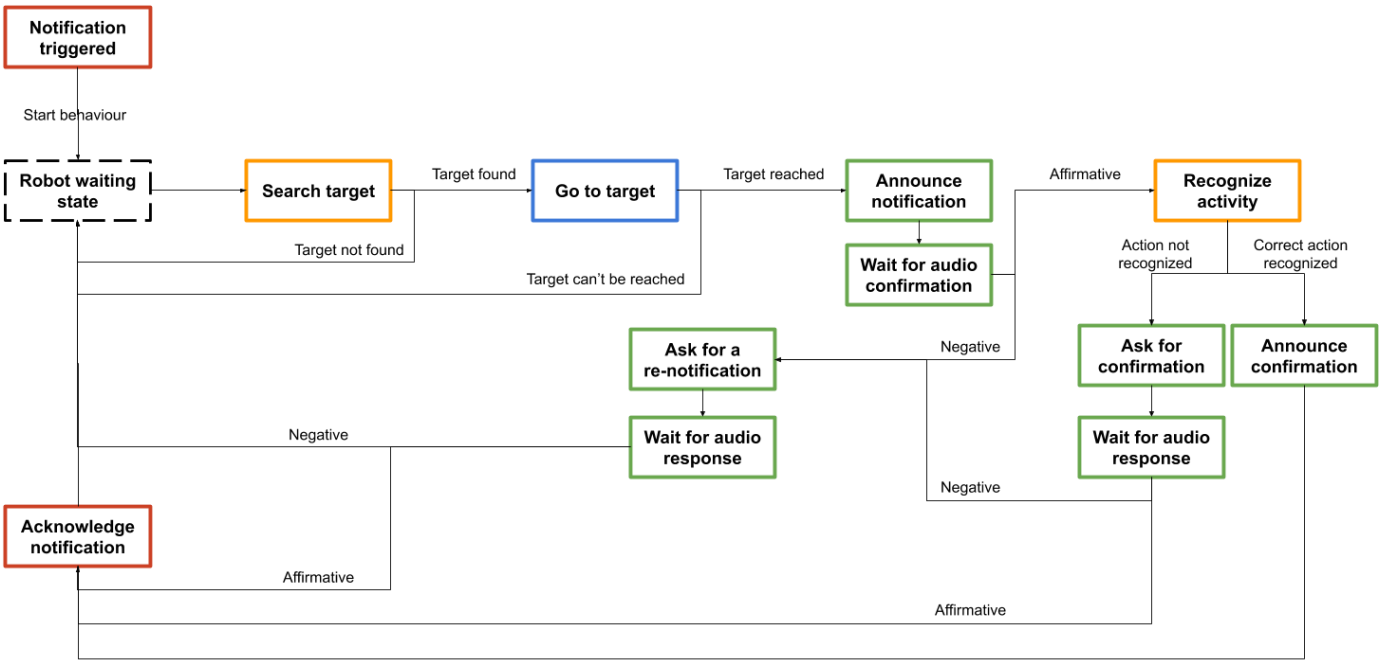


*Figura 2.2. Schema de interacțiune dintre sarcinile incluse în scenariul propus*

O execuție corectă a scenariului poate fi observată în videoclip și acest comportament complex a fost relativ ușor de configurat folosind capabilitățile platformei AMIRO. Schimbul de informații între module funcționează corect, robotul fiind capabil să detecteze, să recunoască și să localizeze persoana în mediu, având de asemenea o interacțiunea vocală de succes. Timpul de răspuns al modulelor este adecvat pentru o interacțiune semnificativă între om și robot, iar experiența generală a utilizatorului este satisfăcătoare.

Am realizat de asemenea un modul de detectare a emoțiilor utilizatorului. Emoțiile afectează starea persoanei și atitudinea acestuia față de un o conversație sau eveniment și joacă un rol esențial în comunicările cu alte persoane. În plus, exprimarea emoțiilor este esențială pentru sănătatea persoanei. Prin urmare, capacitatea de a recunoaște emoția utilizatorului este foarte importantă și utilă pentru un robot social. Modulul de emoție este compus din două componente: componenta de achiziția de date și

[1] http://aimas.cs.pub.ro/robin/en/rezultate/\#demo

componenta de transformarea de date. Componenta de achiziție de date urmărește fața utilizatorului și este responsabilă de achiziția datelor printr-o camera sau printr-un dispozitiv ce are funcții similare și de trimiterea datelor colectate sub un format video către componenta de transformare de date. Componenta de transformare de date extrage "frame-uri" din videoul primit. "Frame-urile" extrase sunt trimise către API-ul Microsoft Azure Face care le analizează și întoarce un fișier JSON care conține rezultatul analizei.

Modulul de emoție al interfeței a fost evaluat de un grup de 11 persoane în laborator. În timpul testelor au fost evaluate doar patru emoții: neutru, fericire, tristețe și furie. Fiecare utilizator a imitat fiecare emoție de zece ori. În consecință, în total au fost realizate un număr de 440 de interacțiuni, 110 interacțiuni pentru fiecare emoție. Rezultatele sunt enumerate în tabelul 2.1. Rezultatele evaluării pentru recunoașterea emoțiilor sunt satisfăcătoare.

| Emoție | Neutru | Fericire | Tristețe | Furie |
|---|---|---|---|---|
| Detectare adevărată | 99 | 102 | 91 | 89 |
| Detectarea falsă | 11 | 8 | 19 | 21 |
| Procent detecție (%) | 90.00% | 92.73% | 82.73% | 80.91% |

*Tabel 2.1 Rezultatele evaluării modulului de emoție.*

Rezultatele au fost sintetizate în lucrarea The AMIRO Social Robotics Framework: Deployment and Evaluation on the Pepper Robot (A.S. Ghita, A. Gavril, M. Nan, B. Hoteit, A. Awada, A. Sorici, I. Mocanu, A.M. Florea) în curs de evaluare la revista Sensors, dar și în lucrarea. Component-Based System Architecture for a Social Service Robot (B. Hoteit, A. Sorici, A.M. Florea, A. Abdallah). Proceedings of the 14th Annual International Technology, Education and Development Conference (INTED), Valencia, Spain, 2020.

# 3. Proiectul component ROBIN-Car

**Parteneri ai proiectului ROBIN-Car**

INSTITUTUL DE MATEMATICĂ "SIMION STOILOW" AL ACADEMIEI ROMANE (CO)
UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI
INSTITUTUL DE CERCETĂRI PENTRU INTELIGENȚĂ ARTIFICIALĂ „MIHAI DRĂGĂNESCU"
UNIVERSITATEA "DUNAREA DE JOS" DIN GALAȚI

## 3.1 Pilotaj automat al mașinii și seturi de date proprii

Am abordat 2 variante de pilotaj automat. Una este bazată strict pe vedere computațională și cea de a doua folosește atât metode de vedere computațională cât și senzori suplimentari.

Din punct de vedere al primului model am realizat un sistem care învață simultan să se auto-localizeze și să navigheze spre o destinație planificată numai din informație video, am extins și îmbunătățit modelul de localizare din imagini și l-am adaptat pentru a învăța să localizeze cu precizie în trafic, am modelat traiectorii, ca funcții de timp ale spațiului, care cuprind informații de direcție și viteză pentru șapte secunde în viitor, din care se pot obține informații complete despre direcție și viteză pe întreaga durată de timp. Traiectoriile captează informații de navigație pe mai multe niveluri, de la comenzi instantanee care depind de traficul actual și obstacolele din față, până la decizii de navigare pe termen mai lung, spre o destinație specifică.

Prezentarea modelului nostru de localizare si navigare vizuala automata, a bazei de date UED precum si prezentarea si discutia in detaliului a rezultatelor noastre de top, in comparative cu modele state of the art din literature, au fost submise recent la jurnalul Sensors (Q1), Special Issue on Sensors and Computer

Vision Techniques for 3D Object Modeling, 2020 (Driven by Vision: Learning Navigation by Visual Localization and Trajectory Prediction, Marius Leordeanu și Iulia Paraicu).

Am extins setul de date propriu Urban European Driving Dataset (UED - setul de date acoperă 350 km conduși în București pe care i-am adnotat cu manevre auto folosind metode automate și nesupervizate, pentru eficientizarea întregului proces.

Din punct de vedere al celei de a 2-a soluții am echipat un autovehicul Logan electric cu 3 camere RGB, 1 smartphone, 1 CAN bus monitor (colecteză date de la senzorii onboard) și un laptop (figura 3.1). Am realizat un model pentru a prezice viteza și unghiul de rotire al volanului pe baza datelor colectate. De asemenea am extins setul de date propriu colectat in campusul UPB (setul de date POLI – figura 3.2), am evaluat și am comparat rezultatele obținute cu cele pentru diferite alte setudi de date disponibile în domeniul public. Rezulatele au fost publicate în lucrarea Neural Road Semantic Segmentation in Driving Scenarios (D. Iancu, A. Sorici, A.M. Florea), ECAI 2020. International Conference on Electronics, Computers and Artificial Intelligence, Romania, 2020. IEEE.

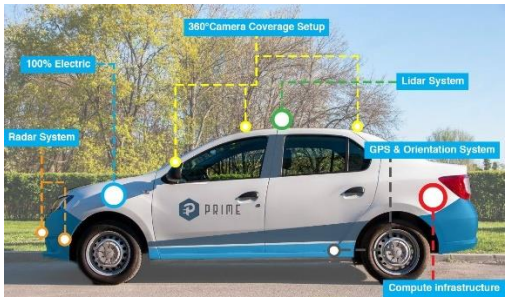


*Figura 3.1 Echipare mașină autonomă*

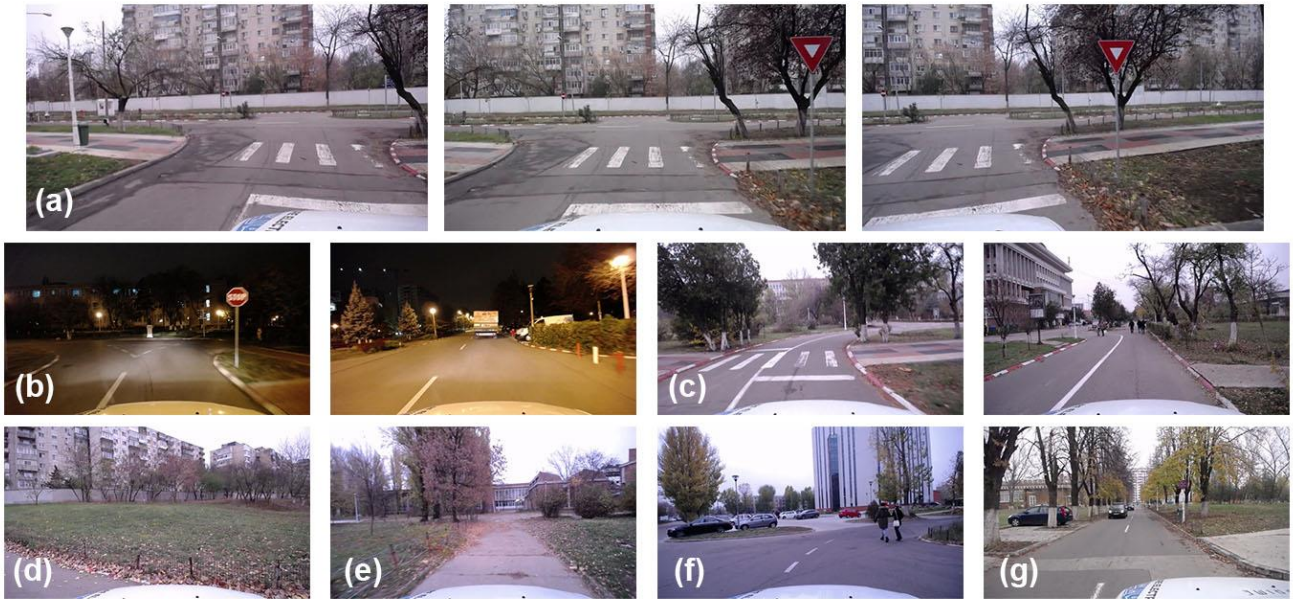


*Figura 3.2 Setul de date POLI*

## 3.2 Descoperirea de obiecte pentru conducere autonomă

Am realizat descoperirea de obiecte în secvențe video (sub formă de măști de obiect pe cadru) într-o manieră nesupervizată. Introducem o metodă eficientă, potrivită pentru scenarii nesupervizate și supervizate, bazată pe o reprezentare a grafului în spațiu și timp, în care avem corespondențe unu-la-unu între nodurile grafului și pixelii video (figura 3.3). Oferim o formulare de clustering spectrală și exploatăm consistența spațio-temporală a obiectului în ceea ce privește mișcarea și aspectul. Nodurile grafice care aparțin obiectului principal de interes ar trebui să formeze un cluster puternic, deoarece sunt legate prin lanțuri de flux optic de lungă durată și au caracteristici de mișcare și aspect similare de-a lungul acestor lanțuri. Pe de o parte, problema de optimizare își propune să maximizeze scorul de

clustering de segmentare pe baza structurii de mișcare prin spațiu și timp. Pe de altă parte, segmentarea ar trebui să fie în concordanță cu privire la caracteristicile nodului. Formularea noastră conduce la o soluție spectrală, anume principalul vector propriu al unei noi matrice Feature-Motion care reprezintă segmentarea obiectelor video atât în spațiu cât și în timp. Deși matricea nu este construită în mod explicit, algoritmul, denumit GO-VOS, calculează eficient vectorul său propriu principal în câțiva pași de iterație. Soluția optimă la problema găsirii valorii proprii asigură că nu se depinde de inițializare și că se găsește obiectul principal fiind cel mai puternic cluster din grafic. În practică, GO-VOS obține rezultate state-of-the-art pe trei seturi de date provocatoare din literatura actuală: DAVIS, SegTrack și YouTube-Objects.

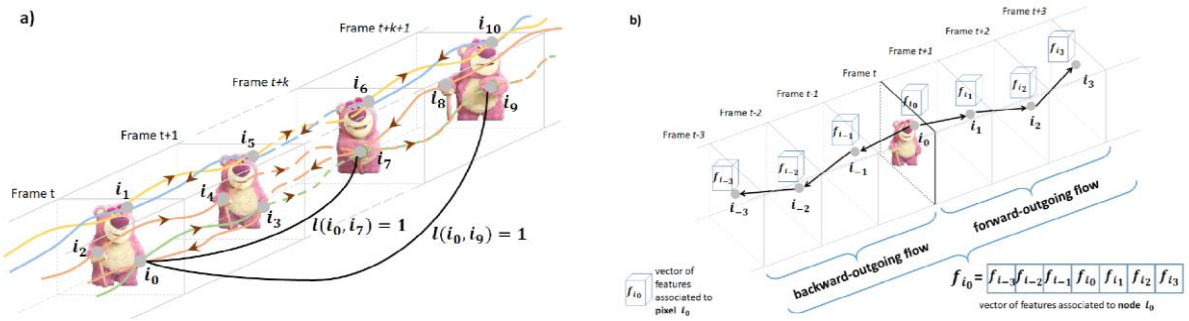


***Figura 3.3:*** *Reprezentare vizuală a modulului de formare a grafului în spațiu-timp*

Lucrarea Unsupervised video object segmentation by reaching Consensus between Deep Object Segmentation and Spectral Spacetime Clustering (E. Haller, A.M. Florea, M. Leordeanu) trimisă la IEEE Trans. on PAMI in 2020, propune un model complet nou de descoperire a obiectelor in care o retea neurala invata de la un graf de descoperire in spatiu si timp, de-a lungul mai multor iteratii sa segmenteze obiectul principal din secvente video. Modelul nou este prezentat in figura 3.4.

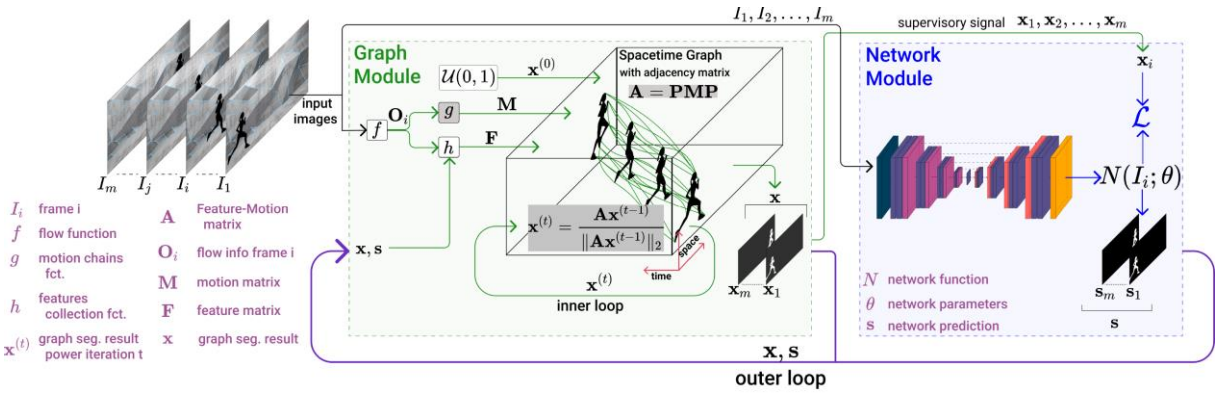


*Figura 3.4: Arhitectura sistemului nostru Iterative Knowledge Exchange (IKE). Modulul grafic (stânga) și modulul rețea (dreapta), schimba informații de-a lungul mai multor cicluri (bucle exterioare) până la convergența. Modulul grafic descoperă obiectul principal din videoclip ca fiind cel mai puternic cluster al grafului spațiu-timp cu corespondențe unu-la-unu între nodurile lui și pixelii video. Soluția de segmentare este vectorul propriu principal al unei noi matrice Feature-Motion (A) calculată eficient cu un algoritm de iterație a puterii proiectat. Apoi, modulul de rețea este antrenat de la zero, folosind rezultatul segmentării grafului ca adnotare nesupervizata, de psedo-adevăr. Reprezentările profunde calculate, împreună cu segmentarea grafului rezultata (x) sunt utilizate pentru a completa caracteristicile la nivel de nod ale grafului și pentru a reinitializa modelul grafic spațiu-timp pentru un nou ciclu (buclă exterioară). Functiile de culoare gri (g, ecuația iterației puterii și calculul matricei de adiacență A) sunt doar entități conceptuale, deoarece algoritmul real evită calculul explicit M și A, reducând drastic sarcina de calcul și memoria utilizate. În lucrare oferim garanții teoretice că algoritmul nostru este echivalent cu iterația de putere si converge într-adevăr la vectorul propriu principal al lui A, care este soluția optimă a problemei de segmentare relaxate. În teste ample, aratam ca sistemul IKE propus crește semnificativ performanța sa pe parcursul mai multor cicluri.*

## Descrierea sistemului de dialog în limbaj natural pentru asistența șoferului

S-a dezvoltat dea semenea, pe baza rezultatelor din proiectul ROBIN-Dialog, un sistem de asistență al șoferului bazat pe dialog în limbja natural. Sistemul pentru asistența șoferului poate interacționa atât cu persoanele din mașină prin interacțiune verbală (persoanele din mașină rostesc întrebări iar sistemul răspunde la acestea tot verbal) cât și cu computerul mașinii (trimite comenzi mașinii pentru a fi executate și primește comenzi verbale din partea acesteia – figura 3.5).

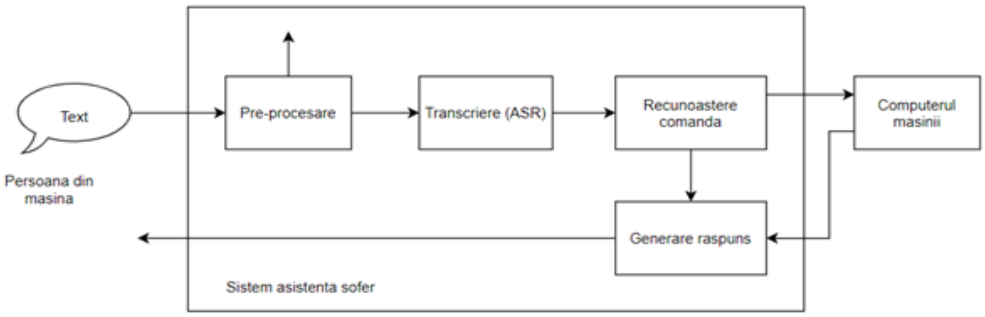


*Figura 3.5. Schema bloc a sistemului de dialog pentru asistența șoferului*

Principalele blocuri ale sistemului pentru asistența șoferului sunt:

- **Blocul de preprocesare**: are rolul de a verifica dacă semnalul audio captat este suficient de puternic pentru a putea fi considerat o cerere (evitând astfel activarea sistemului atunci când persoanele din mașină discută între ele, de exemplu). Semnalele audio slabe nu trec de blocul de preprocesare, prelucrarea lor oprindu-se aici. În schimb, semnalele audio puternice trec și sunt filtrate înainte de a fi trimise mai departe. Filtrarea are loc deoarece mediul din mașină poate fi unul zgomotos: discuții pe fundal, muzică pe fundal, etc.

- **Blocul de transcriere**: Același modul de detecție automată a vorbirii (ASR) folosit și pentru ROBIN Dialog.

- **Blocul de recunoaștere a comenzii**: similar cu managerul de dialog, blocul de recunoaștere identifică ținta comenzii (conceptul) și modul în care se dorește acționat asupra ei (pornire/oprire). De exemplu, următoarele texte sunt echivalente: „Pornește radioul!”/„Deschide radioul!” sau „Pornește faza scurtă!”/„Aprinde luminile de întâlnire!”. Dacă se găsește o comandă corectă, ea va fi trimisă către calculatorul mașinii pentru executare, altfel sistemul va genera singur un răspuns de clarificare pentru a cere persoanei să repete. Răspunsul de clarificare nu obligă utilizatorul să spună ceva, ci doar îl atenționează asupra faptului că sistemul de asistență nu a putut să recunoască o comandă (întrucât există riscul ca anumite semnale audio să treacă în mod incorect de blocul de preprocesare).

- **Blocul pentru generarea răspunsurilor**: redă unul sau mai multe fișiere audio preînregistrate. Poate fi apelat intern de către blocul de recunoaștere a comenzii sau extern (de către computerul mașinii). În urma primirii unei comenzi, computerul mașinii poate trimite înapoi un răspuns pozitiv („Luminile de avarie au fost pornite!”), un răspuns negativ („Radioul este deja închis.”) sau un răspuns general de eroare (pentru a permite șoferului să verifice problema apărută, de exemplu s-a ars un far). Computerul mașinii poate genera răspunsuri fără să fie nevoie să primească o comandă. De exemplu, dacă modul de vedere artificială identifică pietoni pe partea dreaptă, computerul mașinii poate genera răspunsul compus „pieton”, „pe dreapta”. În mod similar se pot genera răspunsuri compuse pentru a descrie și alte rezultate ale modului de vedere artificială.

# 4. Proiectul component ROBIN-Context

**Parteneri ai proiectului ROBIN-Context**

UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI (CO)
UNIVERSITATEA TEHNICĂ DIN CLUJ – NAPOCA

## 4.1 Reprezentarea informatiilor de la obiecte inteligente si conectate la web

Pentru a putea progresa intr-un mod facil cu integrarea framework-ului ROBIN Context in scenariile prevazute pentru aplicatiile din ROBIN Social si ROBIN Car, este nevoie ca informatia ce descrie capabilitatile (proprietati, actiuni, evenimente) ale unui obiect inteligent (eng. web enabled smart thing) sa poata fi usor transformata (serializata / deserializata) din reprezentari tipice programarii orientate obiect in reprezentari interschimbabile pe web: e.g. JSON-LD sau RDF. Pentru reprezentarea informatiilor de context ale unui web-enabled smart thing exista un standar propus de W3C, anume Thing Description (TD)[2]. Un Thing Descriptions descrie metadatele și interfețele acestorThings-uri, în care un Thing este o abstractizare a unei entități fizice sau virtuale care oferă atat interacțiuni cu Web of Things cat și participă la acestea.TD oferă un set de interacțiuni bazate pe un vocabular mic careface posibilă atât integrarea diverselor dispozitive, cât și permite interoperarea diverselor aplicații. În mod implicit, descrierile sunt codificate într-un format JSON care permite, de asemenea, procesarea JSON-LD. Acesta din urmă oferă o bază puternică pentru a reprezenta cunoștințele despre Things într-un mod inteligibil. Un TD poate fi găzduit de Thing în sine sau găzduit extern atunci când acesta are restricții de resurse (de exemplu, spațiu de memorie limitat) sau când un dispozitiv compatibil cu Web of Things este actualizat cuunTD.

**Implementare mecanism de serializare/deserializare**

Framework-ul ROBIN Context are o implementare bazata pe limbajul de programare Python. Din cunostintele noastre, nu exista momentat un framework care sa premita crearea si gestionarea programatica a unor obiecte inteligente - thing-uri - precum si transmiterea informatiilor despre un Thing prin protocoale web (e.g. JSO-LD, RDF). Desi exista unele implementari[3] acestea nu respecta intru totul standardul recomandat de W3C, in special in ceea ce priveste modul de serializare a informatiilor unui Thing in JSON-LD. In plus, nu exista (din cate stim) nicio biblioteca suport care sa permita serializarea unui Thing in format RDF, folosind vocabularul definit de ThingDescription Ontology[4].

Astfel, in cadrul proiectului ROBIN Context se dezvolta o biblioteca de cod prin care se pot defini obiecte inteligente care corespund modelarii ThingDescription. Modelarea acestora poate fi atat transformata din reprezentare Python in reprezentare JSON-LD sau RDF, cat si invers - o reprezentare serializata poate fi transformata intr-un obiect Python utilizabil in programarea interactiunilor sensibile la context din cadrul framework-ului ROBIN Context. Tehnologiile de la care se pleaca in aceasta implementare sunt Webthings, RDFLib, Owlready2[5].

Rezultatele curente ale implementarii sunt public accesibile pe Github[6].

## 4.2 Dezvoltarea rationamentelor probabilistice in platforma ROBIN-Context

In raportul anterior am mentionat ca, in ce priveste adaugarea de rationament probabilistic in platforma ROBIN Context, atentia cade pe procese de clasificare, folosind tehnici de invatare automata, a activitatilor cotidiente (eng. Activities of Daily Living - ADL). Motivul principal este ca acest tip de predictii sunt utile in cadrul proiectului ROBIN Social. In etapa anterioara era subliniat faptul ca detectia activitatilor se face pe baza unor activari de senzori ambientali precum: senzori de miscare, senzori de detectie a inchiderii/deschiderii de usa, senzori de detectie a utilizarii unor robineti sau de folosire a unor

[2] https://www.w3.org/TR/wot-thing-description/
[3] https://iot.mozilla.org/framework/
[4] https://www.w3.org/2019/wot/td
[5] https://pythonhosted.org/Owlready2/
[6] https://github.com/aimas-upb/CONSERT-WoT-Search

obiecte. Astfel, problema principala se rezuma la clasificarea unei serii de activari a acestor senzori intr-una sau mai multe activitatati corespunzatoare acelor activari.

Intr-o prima faza am evaluat capacitatea de clasificare a unor algoritmi tipici de invatare automata (e.g. SVM cu kernel liniar) luand in calcul o impartire exacta pe ferestre de activari (i.e. subsecvente de o lungime data, din sirul activarilor de senzori ambientali), care corespund unei singure activitati. Acuratetea obtinuta era buna pentru majoritatea activitatilor. Cu toate acestea, impartirea exacta pe ferestre de activari nu este un pas care poate fi pus in practica in timp real. Astfel, in continuarea muncii incepute, ne-am orientat catre a evalua capabilitatea de clasificare a activitatii corecte atunci cand ferestrele nu sunt ideale. Am efectuat evaluarea, in continuare, pe setul de date CASAS, prezentat in raportul anterior, adaugand de data aceasta si un alt subset de date: Kyoto - Interleaved ADL Activities[7], HH101 - HH108[8].

**Kyoto.** Setul de date de la Kyoto contine date colectate de la senzori dintr-un apartament inteligent. Apartamentul a fost locuit pe rând de catre 20 de rezidenti care au efectuat 8 activitati în doua moduri:

- secvential - fiecare persoana executa cele 8 activitati intr-o ordine prestabilita
- intercalat - fiecare persoana executa cele 8 activitati intr-o ordine aleatoare,existând posibilitatea de a executa mai multe activitati simultan

Senzorii prezenti în apartament sunt de mai multe tipuri: senzori de miscare, senzori de temperatura sau senzori atasati unor anumite obiecte si sunt distribuiti astfel încât sa acopere o suprafata cât mai mare din apartament.

**HH.** Datele din acest set au fost colectate din 30 de apartamente diferite care erau locuite de catre o singura persoana. Pentru fiecare apartament exista câte un fisier HH, începând cu HH101 pana la HH130. Senzorii care au fost amplasati în apartamentesunt similari cu cei prezenti în setul de date Kyoto.

**Kyoto vs HH.** Fiecare dintre cele doua seturi de date au o serie de particularitati care pot influenta rezultatul testarii. Astfel, pentru setul de date Kyoto sunt luate în considerare numai fisierele care contin datele obtinute de fiecare persoana în mod aleatoriu si eventual si intercalat. De asemenea, pentru acest set de date este folosit un singur aparta-ment, ceea ce înseamna ca toti rezidentii executa activitati in acelasi mediu si prinurmare toti beneficiaza de aceeasi distributie de senzori. În ceea ce priveste seturile de date HH acest lucru se schimba deoarece fiind 30 deapartamente diferite este imposibil sa se mentina aceeasi distributie a senzorilor.

### 4.2.1 Analiza pe setul de date Kyoto

Algoritmul de invatare automata folosit in analiza ferestrelor aleatoare (de lungime 30 - i.e. 30 de activari de senzori) este cel al *padurilor aleatoare de arbori de decizie (eng. Random Forrest).* Setul de atribute extras pentru o astfel de fereastra cuprinde informatie precum: de cel mai frecvent senzor care se activeaza, locul dominant (o impartire logica a pozitiilor spatiale din mediul in care se desfasoara activitatea - e.g. camere dintr-o casa), numar de activari al fiecarui tip de senzor, timpul intre activari, durata ferestrei si altele. Pentru setul de date Kyoto au fost luate în considerare numai fisierele fiecarei persoane care contineau activari de senzori intercalate. Astfel, numarul total de fisiere folosite a fost 20, unul pentru fiecare persoana. Pentru a obtine o evaluare cât mai precisa pe baza setului de date, au fost efectuate doua tipuri de evaluari:

- prima evaluare a constat în efectuarea antrenarii pe 19 persoane urmând ca testarea sa fie efectuata pe cea de-a 20-a persoana. Acest procedeu a fost repetat de 20 de ori astfel încât sa se realizeze o testare pentru fiecare persoanaîn parte
- cea de-a doua evaluare a constat în efectuarea antrenarii pe toate cele 20 depersoane si apoi

[7] G. Singla, D. Cook, and M. Schmitter-Edgecombe, Trackingactivities in complex settings using smart environment technologies. Internatio-nal Journal of BioSciences, Psychiatry and Technology, 1(1):25-35, 2009

[8] D. Cook, A. Crandall, B. Thomas, and N. Krishnan. CASAS:A smart home in a box. IEEE Computer, 46(7):62-69, 2013.

testarea s-a efectuat pentru fiecare persoana pe rând

Rezultatele obtinute[9] constau într-o acuratete care variaza între 0.44 si 0.78 si un F1 score care variaza între 0.40 si 0.61 în functie de tehnicade segmentare în ferestre care a fost folosita. Acestea reprezinta valorile de referinta care vor fi luate în considerare în analizarea rezultatelor obtinute pe cele douaseturi de date prezentate anterior. Aceste rezultate au fost obtinute folosind un set de date colectate din 3 apartamente pe o durata de 6 luni, iar activitatile executate de catre rezidentul fiecarui apartament au fost diferite de cele prezente în seturilede date Kyoto si HH.

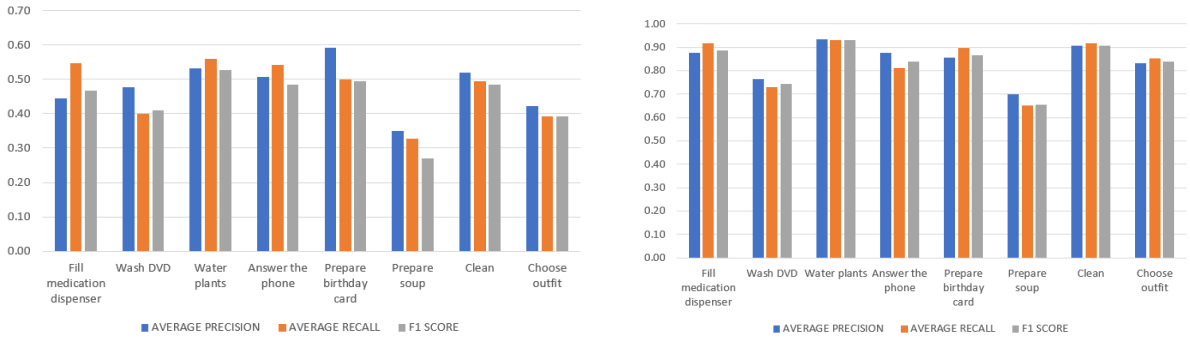


*Figura 4.1 Figura 4.1a (stanga) media rezultatelor algoritmului de tip random forrest pe setul de date Kyoto in antrenarea pe 19 persoane si testarea pe o a 20-a. Figura 4.1b (dreapta) rezultatul obtinut de aceeasi metoda pe acelasi set de date in antrenarea si evaluarea pe acelasi set de date (i.e. acuratetea de anternare)*

Rezultatele evaluarii folosind 19 persoane pentru setul de antrenare si cea de-a 20-a pentru testare sunt prezentate în Figura 4.2a. Rezultatul reprezinta media rezultatelor obtinute de fiecare persoana. Observam ca rezultatele sunt similare cu cele de referinta având o precizie medie pentru toate activitatile de 0.48, unrecall mediu de 0.47 si un scor F1 mediu de 0.44. Folosind cea de-a doua metoda de evaluare, sianume folosirea întregului set de date pentru antrenare si apoi testarea pentru fie-care persoana în parte pe acelasi model creat, aduce rezultate mult mai bune decât prima metoda încercata. Astfel, am realizat din nou media rezultatelor obtinute pen-tru fiecare persoana în parte (Figura 4.2b) si s-au obtinut o precizie medie si unrecallmediu de 0.84, iar scorul F1 mediu a fost 0.83.
Un rezultat important observat din analiza setului de date Kyoto este ca gradul de generalizare de la activitățile executate de o persoana la o alta este scăzut. Este asadar nevoie de o dezvoltare a unor tehnici de fine-tuning ale algoritmului de recunoastere a activitatilor pe baza clasificarii de ferestre de activari de senzori, astfel incat acesta sa poată fi particularizat pe fiecare persoana în parte.

### 4.2.3 Analiza pe setul de date HH

Analiza a fost executata asupra primelor 8 fisere din setul de date, de la HH101 pâna la HH108. Aceasta evaluare a avut un grad de complexitate mai mare decât în cazul setului de date Kyoto. Astfel, fisierul principal cu date a fost împartit în mai multe fisiere mai mici, fiecare dintre acestea continând activari de senzori dintr-o anumita zi. Asadar, toate activarile de senzori dintr-o zi se regaseau în acelasi fisier. Urmatorul pas a fost împartirea fisierelor în seturi de câte 6, urmând sa fie folosite primele 4 fisiere pentru antrenare si restul de 2 pentru testare. Glisarea ferestrei de cate 6 zile cu cate o zi asigura faptul ca nu se induc bias-uri peste zilele din saptamana care fac parte din setul de date de antrenare.
Pentru fiecare din cele 4 evaluari ale fiecarui bloc de 6 fisiere s-a obtinut o acuratete si pe baza acestora s-a calculat o acuratete medie pentru fiecare bloc în parte. Ulterior, s-a calculat media acuratetilor pentru toate blocurile de 6 fisiere, rezultând astfel o acuratete generala pentru întreg setul de date.
Rezultatele obtinute sunt sub nivelul celor de referinta, dar sunt si sub nivelul celorobtinute pe setul de date Kyoto. Astfel, acuratetea a variat între 0.37 si 0.5 dupa cum se poate observa în Figura 4.3.
Consideram ca modul fix (secventa de 30 de activari de senzori) in care se calculeaza ferestrele de evenimente pe baza carora se extrag atributele utilizate in algoritmul Random Forest este inadecvat.

[9] Krishnan, N. C., Cook, D. J. (2014). Activity recognition onstreaming sensor data. Pervasive and mobile computing, 10, 138-154

Este nevoie asadar de îmbunatatirea segmentarii datelor în ferestre prin identificarea *momentelor* în care se trece de la o activitate la alta. Astfel, identificând aceste puncte se poate obtine o segmentare mult mai precisa a datelor deoarece, daca punctele detectate sunt corecte, înseamna ca între doua puncte este desfasurata exact o activitate. Asadar, dificultatea acestei abordari consta în identificarea cât mai corecta a acestor puncte în care se trece de la oactivitate la alta. Implementarea unor metode de detectie a punctelor de schimbare (eng. change point detection) face obiectul continuarii lucrarilor.

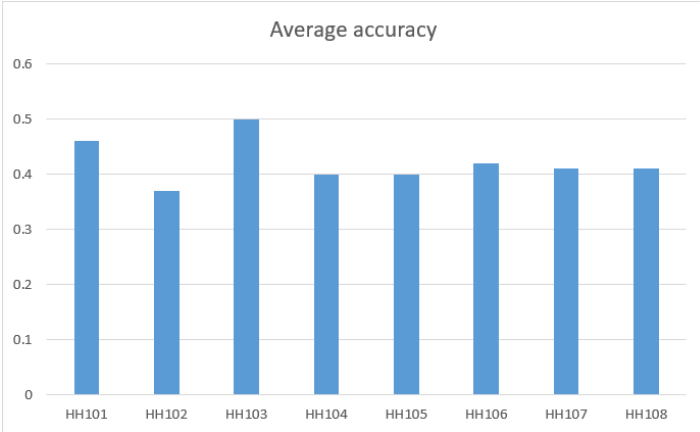


*Figura 4.2. Grafic de acuratete in predictia activitatilor desfasurate de rezidentii celor 8 apartamente din setul de date CASAS HH.*

Pentru finalizarea etapei curente a ROBIN Context efortul de cercetare si dezvoltare se va concentra pe urmatoarele:

- Definitivarea framework-ului de serializare/deserializare pentru ThingDescription
- Utilizarea framework-ului ThingDescription pentru implementarea integrarii platformei ROBIN Context ca furnizor de informatie si mediator de actuare asupra elementelor dinamice din scenariile demonstrative ale proiectelor componente ROBIN Social si ROBIN Car. Aceasta presupune:
  - Implementarea conceptelor de ContextDomainGroup si a agentilor care il gestioneaza (elemente mentionate in raportul anterior) ca thing-uri virtuale
  - Utilizarea agentilor si a ContextDomainGroups pentru a raspunde la nevoie de acces si partajare a informatiei de context in aplicatiile din ROBIN Social si ROBIN Car
- Finalizarea dezvoltarii suportului pentru inferenta de informatie de context privitoare la activitatea umana utilizand tehnici de invatara automata. In mod particular va fi implementata o metoda hibrida de extragere a punctelor de schimbare (eng. change points) astfel incat ferestrele pe care se face clasificarea evenimentelor de la senzori sa aiba o probabilitate cat mai mare de a apartine unei singure activitati.

# 5. Proiectul component ROBIN-Dialog

**Parteneri ai proiectului ROBIN-Dialog**

INSTITUTUL DE CERCETĂRI PENTRU INTELIGENȚĂ ARTIFICIALĂ „MIHAI DRĂGĂNESCU" (CO)
UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI
UNIVERSITATEA TEHNICĂ DIN CLUJ – NAPOCA

## 5.1 Prezentare generală

În anul 2020 activitatea de cercetare a fost dublată de eforturile de dezvoltarea a unei noi soluții de ASR cu performanțe mai bune decât a variantei anterioare. Managerul de dialog a fost îmbunătățit și el substanțial prin extensia capacității de înțelegere a unor enunțuri eliptice precum și adaptarea la noua soluție de ASR. Au fost dezvoltate noi micro-lumi conform unor scenarii suplimentare. Noile dezvoltări

sunt disponibile în github[10], clonabil și în platforma ERRIS. În plus, platforma publica RELATE a fost îmbogățită cu noul modul ASR și un modul de înregistrare a vorbirii, astfel încât utilizatorii să poată experimenta direct în browser noile facilități.

Pentru implementarea sistemului de dialog în limbaj natural pentru micro-lumea unui robot asistiv/teleprezență au fost elaborate încă două microlumi (una la ICIA și UPB și alta la UTCN) pentru un robot asistiv în două ipostaze:

a) asistent casnic, pentru care robotul poate susține dialoguri referitoare la interogări/comenzi de genul: aprinde/stinge lumina, crește/scade/setează temperatura în cameră, pornește/oprește muzica/tv, adaugă/întreabă eveniment în calendar. A fost implementat un flux de prelucrare bazat pe RASA cu un manager de dialog implementat în Prolog. Pentru partea de prelucrare a vocii au fost folosite aplicațiile Google Speech. Contribuția este a partenerilor de la UTCN.

b) o persoană cu dizabilități sau în vârstă, pentru care au fost schițate 5 scenarii. Scenariile au fost documentate în fișiere de tip mw (microworld) la adresa src\main\resources\asistiv.mw din GitHub și pot fi utilizate cu managerul de dialog RDM.

## 5.2 Descrierea scenariului Asistent Casnic

Un asistent casnic este un sistem conversațional inteligent care este capabil sa asiste utilizatorul în automatizarea diferitelor acțiuni ce țin de diferiții parametri ai casei (temperatura, lumina, etc), sau răspunde diferitelor interogări pe care utilizatorul le-ar putea efectua (e.g. întrebări despre vreme, planificarea activității zilnice, etc).

Pentru scenariul de asistent casnic, am definit mai multe tipuri de interogări/comenzi, intențiile și parametrii asociați acestora fiind prezentați în tabelul 5.1.

| **Nume intentie (RO)** | **Parametri** | **Categorie** | **Tip** |
|---|---|---|---|
| *AprindeLumina* | Camera | Lumina | comanda |
| *StingeLumina* | Camera | Lumina | comanda |
| *ScadeIntensitateLumina* | Camera, nivel | Lumina | comanda |
| *CresteIntensitateLumina* | Camera, nivel | Lumina | comanda |
| *CresteTemperatura* | Camera, nivel | Temperatura | comanda |
| *ScadeTemperatura* | Camera, nivel | Temperatura | comanda |
| *SeteazaTemperatura* | Camera, nivel | Temperatura | comanda |
| *PuneMuzica* | Artist | Media | comanda |
| *OpresteMuzica* | - | Media | comanda |
| *CresteIntensitateMuzica* | Nivel | Media | comanda |
| *ScadeIntensitateMuzica* | Nivel | Media | comanda |
| *PornesteTV* | Canal | Media | comanda |
| *OpresteTV* | - | Media | comanda |
| *SchimbaCanal* | Canal | Media | comanda |
| *AdaugaEventCalendar* | Nume, data, ora inceput, durata | Calendar | comanda |
| *IntreabaEventCalendar* | Data, ora inceput, ora final | Calendar | interogare |
| *IntreabaVreme* | Loc, timp | Vreme | interogare |

*Tabel 5.1 - Intențiile definite pentru scenariul de asistent casnic*

Am proiectat un flux de procesare pentru un asemenea sistem, care integrează diferite componente

[10] https://github.com/racai-ai/ROBINDialog

pentru rezolvarea diferitelor sarcini. Figurile 5.1 și 5.2 prezinta fluxul în cele 2 variante: procesarea unei interogări, respectiv a unei comenzi. Figurile arată și deciziile de implementare posibile pentru fiecare componenta. Implementarea componentei de generare de limbaj natural nu a fost momentan abordată.

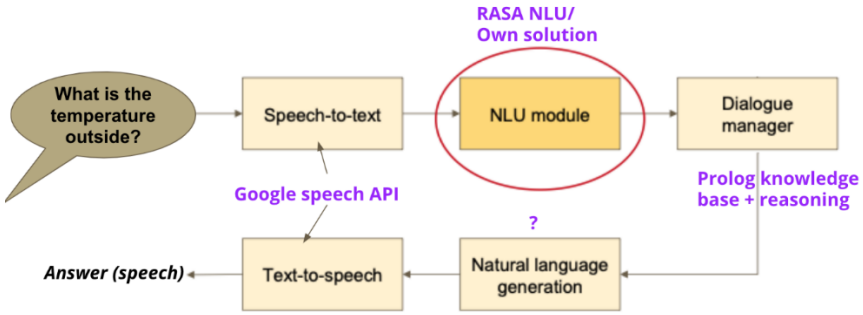


*Figura 5.1. Flux de procesare pentru cereri de tip interogare*

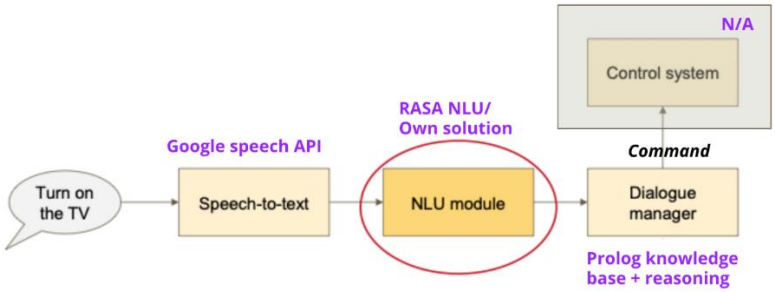


*Figura 5.2. Flux de procesare pentru cereri de tip comandă*

Ne-am concentrat pe implementarea modulelor de NLU (detecția intenției și a parametrilor acesteia), precum și pe implementarea unui prototip de manager de dialog pentru scenariul de asistent casnic.

## 5.3 Descrierea scenariilor pentru un robot asistent al unei persoane cu dizabilități sau în vârstă

Așa cum am arătat în rapoartele anterioare, pentru a putea susține un dialog coerent și cooperant cu utilizatorul, robotului Pepper trebuie sa i se formalizeze conceptele și acțiunile relevante în micro-lumea respectivă. În cele ce urmează este prezentată reprezentarea parțială a micro-lumii asistentului social al unei persoane cu dizabilități sau în vârstă (care poate fi extinsă, în conformitate cu regulile sintactice ale limbajului de definire a micro-lumilor). Dăm aici listingul fișierului `src\main\resources\asistiv.mw` din GitHub. Acest fișier-definiție a unei micro-lumi de robot asistiv dă posibilitatea managerului de dialog RDM să răspundă punctual unor întrebări cum ar fi: „Ce medicament trebuie să iau astăzi?” / „Ce zi este astăzi?” / „Cât e ceasul?” / „Câte grade sunt afară?” / „Trebuie să iau Extraveral astăzi?” / etc.

### 5.3.1 Îmbunătățiri aduse managerului de dialog RDM

Managerul de dialog pentru micro-lumi ROBIN Dialog Manager (RDM, https://github.com/racai-ai/ROBINDialog) a fost descris în lucrarea „A Dialog Manager for Micro-Worlds”[11] și care, de la momentul scrierii până în prezent, a primit următoarele îmbunătățiri:

[11] Ion R., Badea V. G., Cioroiu G., Barbu Mititelu V., Irimia E., Mitrofan M. și Tufiș D. (2020). A Dialog Manager for Micro-Worlds. Studies în Informatics and Control, 29(4) 401-410, December 2020. ISSN: 1220-1766

1. *Referințe ale conceptelor din micro-lume rezolvate prin apeluri către clase Java*. În acest mod, putem răspunde în limba română unor întrebări care necesită un anumit tip de calcul. De exemplu, la întrebarea „Cât este ceasul?”, RDM trebuie să afle ora curentă din sistem și s-o traducă în fraza corespunzătoare în limba română. Acest calcul este implementat în clasa ro/racai/robin/dialog/generators/TimeNow.java. Pentru a preciza referința substantivului comun „oră” în definiția micro-lumii, vom adăuga linia

   ```
   REFERENCE oră ro.racai.robin.dialog.generators.TimeNow = G1
   ```

   împreună cu predicatul `TRUE fi G1`

   Clasa ro/racai/robin/dialog/generators/DegreesNow.java ne permite să răspundem întrebării „Câte grade sunt afară?” sau „Ce temperatură e afară?”, utilizând informații despre locul unde se află RDM (după adresa IP) și serviciul web gratuit pus la dispoziție de Agenția Națională de Meteorologie (http://www.meteoromania.ro/).

2. Modulul de „Text-To-Speech (TTS)” a fost înlocuit cu unul mult mai rapid, până când găsim resurse pe care să instalăm modulul TTS descris în lucrare, bazat pe rețele neuronale, care are o intonație mai apropiată de realitate. Am folosit librăria SSLA (Boroș et al., 2018), scrisă în Java, care sintetizează foarte rapid (sub o secundă) fraze de 10-20 de cuvinte.

3. Modulul de „Automatic Speech Recognition” (ASR) a fost înlocuit cu unul mult mai performant (cu o eroare medie de recunoaștere a cuvintelor de trei ori mai mică decât modulul ASR descris în lucrare) și mai rapid, bazat pe rețeaua neuronală complexă Deep Speech 2 (Avram et al., 2020). În cele ce urmează, vom detalia acest modul de ASR.

### 5.3.2 Modul ASR pentru limba română bazat pe Deep Speech 2

Avram et al. (2020) descriu un modul ASR pentru limba română bazat pe Deep Speech 2. Modulul a fost antrenat pe cca. 230 de corpus bimodal (voce și text) și reușește să obțină o rată medie a erorii de recunoaștere la nivel de cuvânt (eng. „Word Error Rate” sau WER) de 9.9%. Modulul poate transcrie fraze rostite în 70 ms pe frază, un timp de răspuns excelent pentru o aplicație cum e RDM în care utilizatorul așteaptă răspunsul într-un timp rezonabil. Rata de eroare de 9.9% încorporează erori de recunoaștere a folosirii cratimei în limba română pentru a separa clitice sau pentru a lega cuvintele în rostire. Am dezvoltat module speciale de corectură pentru aceste situații, descrise în cele ce urmează.

### 5.3.3 Module de corectură a transcrierii ASR

Modulul de recunoaștere a vorbirii[12] dezvoltat în cadrul proiectului ROBIN, deși are performanțe foarte bune (eroarea la nivel de cuvânt WER sub 10%) pentru texte apropiate celor din setul de antrenament, poate avea dificultăți în adaptarea la un vorbitor necunoscut sau la texte complet diferite celor de antrenament. Având în vedere acest aspect, precum și caracteristicile micro-lumilor investigate în proiectul ROBIN, au fost dezvoltate două module de corecție a textului recunoscut: restaurare cratimă + capitalizare pentru cuvinte cunoscute și corecție cuvinte necunoscute.

Modulele de corectură au fost implementate sub forma unor servicii web REST, expuse prin intermediul protocolului HTTP. Acestea permit transferul textului rezultat în urma procesului de recunoaștere a vorbirii și corectarea acestuia precum și returnarea textului final rezultat către aplicația client. Având în vedere specificul dialogului din proiectul ROBIN (propoziții unice sau texte relativ scurte), interogarea serviciilor web se poate realiza prin intermediul metodei HTTP GET și utilizarea unui parametru din structura URL-ului aferent fiecărui serviciu. Rezultatul întors este sub formă de document JSON. Textul corectat se regăsește în proprietatea ”text”. Totodată pentru a ușura integrarea în aplicații în documentul JSON rezultat se regăsește și un câmp ”status” care va conține valoarea ”OK” dacă procesarea s-a efectuat cu succes. Adițional, JSON-ul rezultat conține un câmp ”comments” care conține diferite informații despre aplicarea modelului și deciziile luate de acesta. Aceste informații putând fi apoi

[12] Avram, A.M., Păiș, V., Tufiș, D. (2020) Towards a Romanian end-to-end automatic speech recognition based on Deep Speech 2. în Proceedings of the Romanian Academy, Series A, in-print

utilizate pentru a înțelege anumite corecții efectuate precum și pentru a investiga eventuale opțiuni pentru îmbunătățirea performanțelor.

### 5.3.4 Modulul pentru restaurarea cratimei și a majusculelor în cuvintele cunoscute

Considerând două forme de scriere (cu sau fără cratimă): ”sau” / ”s-au” acestea au semnificații diferite. Exemplu: ”Cei doi prieteni s-au dus la munte sau la mare.” În primul caz ”s-au” avem un pronume reflexiv ”s-” și un verb auxiliar ”au”, în timp ce în al doilea caz ”sau” este o conjuncție. Astfel, recunoașterea eronată a lui ”s-au” în forma fără cratimă conduce la erori în etapele ulterioare ale procesărilor din cadrul proiectului ROBIN. Modulul de corecție realizat utilizează un model bazat pe bigrame de forma ($W_k$, $W_{k+1}$) pentru a corecta cuvântul curent $W_k$. Pentru a reduce dimensiunea modelului, au fost incluse doar formele $W_k$ care acceptă atât scrierea cu cratimă cât și fără. Acest model a fost antrenat utilizând corpusul CoRoLa[13]. Dacă modelul nu poate identifica un bigram (posibil ca urmare a unei probleme în recunoașterea cuvântului următor), se recurge la un model unigram bazat pe frecvențele de apariție a celor două forme (cu și fără cratimă).

Scrierea corectă, cu majuscule atunci când sunt nume proprii, joacă de asemenea un rol important în procesările ulterioare. Astfel, identificarea părților de vorbire poate utiliza majusculă întâlnită în mijlocul propoziției pentru a identifica un substantiv propriu sau procesele de recunoaștere a entităților pot utiliza majuscula ca factor care contribuie la identificarea unui cuvânt ca parte a unei entități. Pentru a corecta cuvintele și a introduce majuscule acolo unde este cazul, a fost utilizată o listă de cuvinte cunoscute ca fiind asociate entităților cu nume (persoane, locații, organizații). Având în vedere că se urmărește doar corectarea literelor mici în majuscule, nu este relevant dacă același cuvânt poate avea mai multe semnificații (cum ar fi o locație care poate fi utilizată și în cadrul unei entități de tip organizație).

### 5.3.5 Modulul pentru corectarea cuvintelor necunoscute

Proiectul ROBIN se bazează pe existența unor micro-lumi în care este permisă interacțiunea om-robot. Astfel, vocabularul aferent acestor micro-lumi este relativ redus, ceea ce permite presupunerea că un cuvânt care nu există în vocabular este cel mai probabil recunoscut greșit de sistemul de recunoaștere a vorbirii. Astfel, modulul pentru corectarea cuvintelor necunoscute încearcă să înlocuiască orice cuvânt necunoscut cu un cuvânt apropiat care poate fi utilizat corespunzător contextului întâlnit. Se presupune că nu există o întreagă secvență de cuvinte recunoscută eronat.

Au fost construite 2 modele bigram ($W_k$,$W_{k+1}$), ($W_{k-1}$, $W_k$) și un model unigram ($W_k$). Ca și la modulul anterior, modelele au fost antrenate utilizând corpusul CoRoLa. În vederea corectării, la identificarea unui cuvânt $W_k$ necunoscut, se încearcă identificarea posibilelor cuvinte corecte pe baza celor 3 modele. Cuvântul ales corespunde celui care se poate găsi în contextul curent și are distanța Levenshtein cea mai mică. Pentru a evita calculul tuturor distanțelor Levenshtein este impusă o limită ca diferență de dimensiune maximă între cuvântul curent și un potențial candidat.

### 5.3.6 Integrarea modulelor de corectare

Având în vedere specificitatea modulelor pentru proiectul ROBIN, precum și faptul că mecanismul general de recunoaștere a vorbirii (ASR + corecturi) poate fi utilizat în diferite scenarii, este necesară integrarea diferitelor componente în funcție de necesitățile specifice ale aplicației care le utilizează. Este posibilă, astfel, fie utilizarea directă a ieșirii modulului ASR, fie combinarea unuia sau mai multor module de corectură. Schema bloc a unui sistem care realizează integrarea diferitelor module este prezentată în Figura 5.3.

---

[13] Tufiș, D., Mititelu, V.B., Irimia, E., Păiș, V., Ion, R., Diewald, N., Mitrofan, M., Onofrei, M. (2019). Little strokes fell great oaks. Creating CoRoLa, the reference corpus of contemporary Romanian. în Revue Roumaine de linguistique, LXIV.

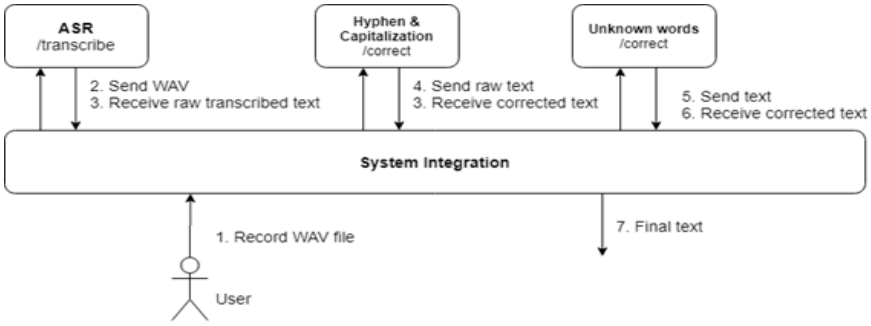


*Figura 5.3. Schema bloc a unui sistem care realizează integrarea sistemului ASR cu modulele de corectură*

În vederea testării acestei integrări, a fost realizată o pagină web demonstrativă în cadrul platformei RELATE[14]. Utilizând platforma, este posibilă încărcarea sau înregistrarea unei propoziții în format audio (fișier WAV). Ulterior, componenta de integrare apelează diferitele module (ASR, corectură cratimă + capitalizare, corectură cuvinte necunoscute), conform diagramei prezentată în Figura 5. Rezultatul final este apoi afișat utilizatorului, fiind posibil apoi transferul automat al acestui rezultat către analiză, utilizând facilitățile oferite de platforma RELATE. În Figura 5.4 este prezentată componenta de interfață aferentă platformei RELATE care permite înregistrarea sunetului în vederea utilizării sistemului de recunoaștere a vorbirii. Ulterior, în Figura 5.5 este prezentat rezultatul recunoașterii vorbirii.

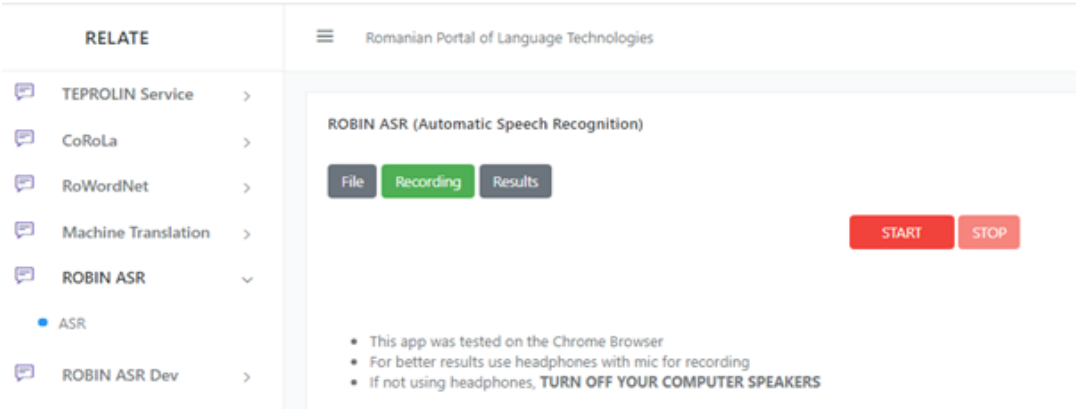


*Figura 5.4. Componenta de interfață din platforma RELATE care permite înregistrarea sunetului și transferul către componenta de recunoaștere a vorbirii*

Componenta de înregistrare a fost documentată cu un manual de utilizare ce arată pas-cu-pas operațiile necesare pe care trebuie să le facă un utilizator pentru a genera un fișier audio și a-l trimite componentei ASR.

Ulterior recunoașterii sunetului și corecturii, în cadrul platformei se poate invoca automat serviciul TEPROLIN[15] care permite analiza textului prin apăsarea butonului „Analysis” din Figura 5.5 (același lucru fiind realizat și intern în cadrul proiectului ROBIN). Prin intermediul afișării rezultatelor în cadrul platformei RELATE, este posibilă verificarea diferitelor nivele de adnotare, precum și explorarea grafică a relațiilor dintre componentele textuale, care permite ajustarea algoritmilor de nivel superior utilizați apoi în cadrul proiectului ROBIN. Figura 5.6 prezintă grafic structura frazei recunoscută de sistemul de recunoaștere a vorbirii.

[14] http://relate.racai.ro

[15] Ion, Radu. (2018). TEPROLIN: An Extensible, Online Text Preprocessing Platform for Romanian. în Proceedings of the International Conference on Linguistic Resources and Tools for Processing Romanian Language (ConsILR 2018), November 22-23, 2018, Iași, România.

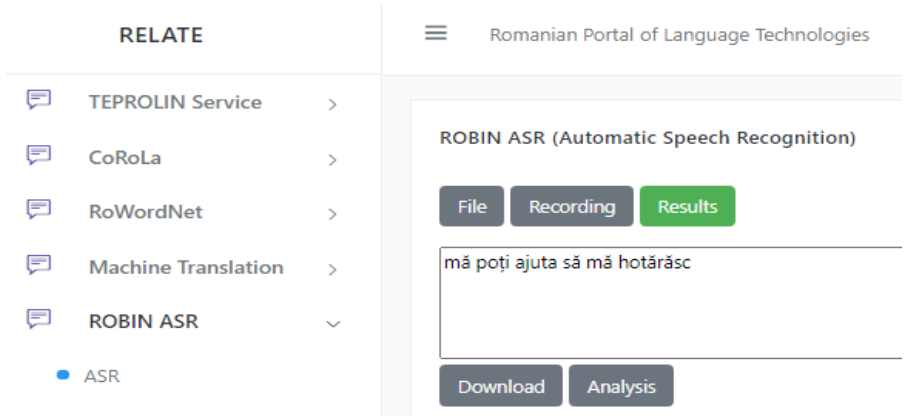


*Figura 5.5. Recunoașterea vorbirii înregistrate prin platforma RELATE ca urmare a trecerii semnalului prin toate modulele de corectură, conform diagramei bloc din Figura 5.4.*

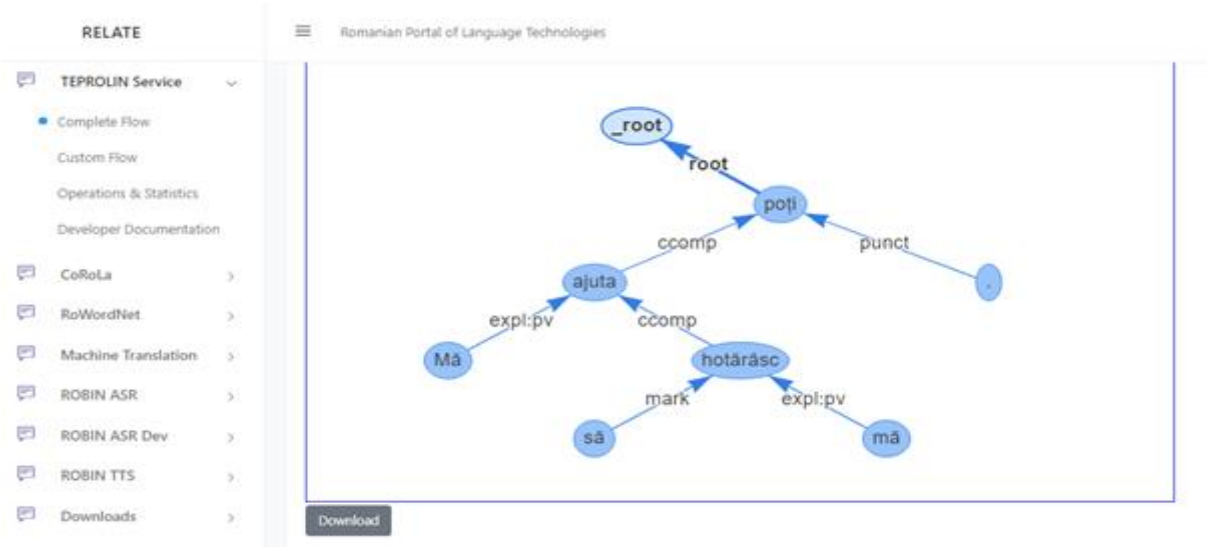


*Figura 5.6. Structura frazei recunoscută de sistemul de recunoaștere a vorbirii*

### 5.3.7 Îmbunătățiri ale modelului ASR

În vederea îmbunătățirii modelului ASR și a adaptării acestuia la specificul micro-lumilor ROBIN, a fost creat un corpus bimodal (text + voce). Acesta este format din 700 de propoziții specifice antrenării unui ajutor în vânzări aferent unui magazin de calculatoare. În vederea înregistrării acestora, a fost dezvoltat un modul nou în platforma RELATE care permite încărcarea unui fișier CSV cu propozițiile și apoi înregistrarea semnalului audio aferent acestora de către utilizatorii platformei.

Pentru a ușura munca celor care înregistrează sunetul și totodată pentru a asigura o aliniere perfectă cu textul, propozițiile din corpus conțin o scriere apropiată de pronunție. Astfel în loc de cuvântul „laptop” a fost utilizată scrierea „leptop” pentru a indica persoanei care înregistrează pronunția dorită. În vederea înregistrării sunetului, au fost înregistrați în platformă 6 utilizatori. Deoarece nu toți utilizatorii vor înregistra integral textele aferente corpusului, s-a realizat o împărțire a acestuia în 4 sub-corpusuri. Fiecare utilizator va începe înregistrările cu un anumit sub-corpus astfel încât la final să fie asigurată existența a cel puțin doi vorbitori pentru fiecare sub-corpus. La finalul înregistrărilor, corpusul urmează să fie anonimizat și îmbogățit cu metadate, descriind cel puțin modalitatea de realizare, caracteristicile tehnice ale înregistrărilor, numărul de vorbitori, caracteristici ale vorbitorilor, etc.

Toate programele și structurile de date aferente lor au fost plasate în github-ul public https://github.com/racai-ai/ROBINDialog, clonabil și în platforma ERRIS. Utilizatorii interesați pot folosi serviciile implementate prin intermediul unui browser accesând platforma RELATE http://relate.racai.ro.

# 6. Proiectul component ROBIN-Cloud

**Parteneri ai proiectului ROBIN-Cloud**

UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI (CO)
UNIVERSITATEA TEHNICĂ DIN CLUJ – NAPOCA
UNIVERSITATEA "DUNAREA DE JOS" DIN GALAȚI

## 6.1 Planificare inteligentă și adaptivă

Proiectul ROBIN-Cloud este definit în contextul în care majoritatea bibliotecilor ce implementează algoritmi de procesare / învățare pe volume mari de date sau pe serii temporale nu oferă suport pentru prelucrare paralelă și/sau distribuită, motiv pentru care tot suportul de scalabilitate oferit de un mediu Cloud devine neutilizabil. Algoritmi de Machine Learning, Deep Learning mai nou, sau Computer Vision, nu se pot implementa ușor în platforme gen MapReduce, Hadoop, Apache Spark, Apache Flink, sau pe modele de algoritmi bazați pe fluxuri de procesare cu dependențe. O altă problemă constă în lipsa interoperabilității la nivelul datelor, care provin din surse diverse, senzori dezvoltați în tehnologii diferite.

Realizările proiectului se vor prezenta relativ la aceste obiective. Vom prezenta extinderea principalelor funcționalități Cloud printr-un concept nou și general de micro-servicii, oferind suport pentru colectare, stocare, procesare și regăsire rapidă de cunoștințe / învățare, pentru proiectele definite în cadrul ROBIN.

Pornind de la un proiect bine cunoscut Cloud-Edge, în care puterea de procesare și stocarea sunt oferite de Cloud pentru a ajuta dispozitivele edge, îl extindem cu un alt nivel, numit Fog, care a fost folosit și în literatura de specialitate. În cazul de utilizare generală, procesarea Fog sau Edge se referă la reducerea decalajului dintre resursele Cloud și producătorii de date și consumatori, reprezentate de dispozitivele edge. Folosim acest nivel cu roluri mai bine definite, pentru a supraveghea nodurile edge, a intermedia comunicarea între ele și între Edge și Cloud. Făcând acest lucru, reducem complexitatea nodurilor Edge și obținem posibilitatea de a echilibra volumul de muncă între noduri. O imagine de ansamblu a întregii arhitecturi este prezentată în Figura 6.1.

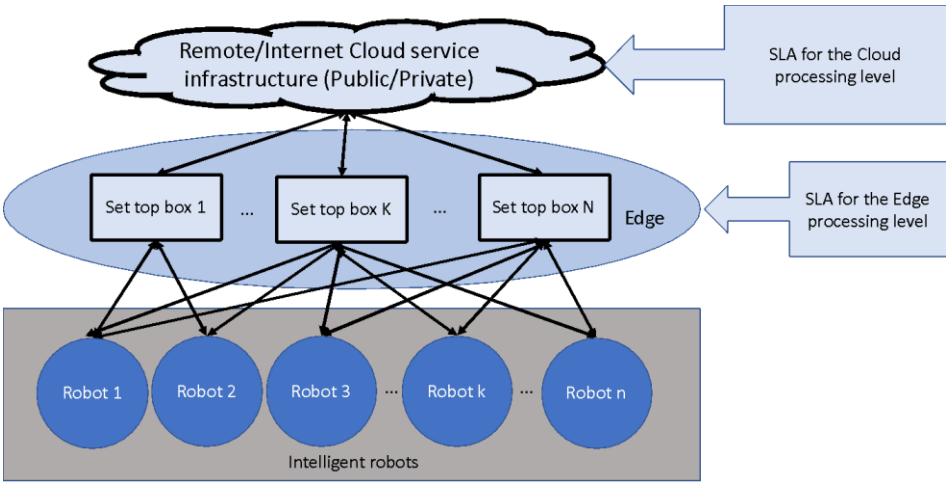


*Figura 6.1. Vederea de ansamblu a procesării Edge-Fog-Cloud la nivelul proiectului ROBIN-Cloud*

**Nivelul Edge** este format din noduri care pot aduna date de la mediul ambiant. În modelul prezentat acest tip de nod se numește worker sau lucrător și are următoarele funcții:

- Înregistrare video
- Înregistrare audio
- Urmărirea mișcărilor bazată pe UDS

**Nivelul Fog** este cel responsabil de verificarea resurselor disponibile ale fiecărui lucrător și de a face planificarea sarcinilor. Acesta este format din noduri numite supervizori. Supervizorul primește periodic

mesaje speciale care conțin informații despre resursele pe care fiecare lucrător le are la un moment dat. În plus primește informații despre serviciile disponibile pentru fiecare lucrător. Acest lucru este foarte util pentru algoritmul de planificare a sarcinilor, deoarece nu toți lucrătorii sunt capabili să îndeplinească aceleași servicii, trebuie făcută o filtrare inițială. Mai mult supervizorul primește sarcini care trebuie programate pe un alt dispozitiv. Deoarece nu cunoaște momentul în care un lucrător trimite o solicitare de descărcare a sarcinilor, algoritmul nu poate lua în considerare acest factor. Prin urmare, combină date despre resursele disponibile ale lucrătorilor și o estimare a consumului de resurse pentru fiecare sarcină și alege cel mai bun nod în care fiecare sarcină urmează să fie descărcată.

**Nivelul Cloud**. Întrucât există o cantitate gigantică de date produse și procesate, Cloud servește în principal ca dispozitiv de stocare persistent. Al doilea rol este de a face doar partea de procesare care nu are un rezultat imediat important ce trebuie trimis înapoi la nodurile lucrător din nivelul Edge. Unul dintre principalele servicii Cloud care a fost considerat de la început a fost cel definit de proiectul Robin-Cloud, care își propune să dezvolte o platformă de sprijin pentru roboții IoT. În acest model, resursele de procesare, concepute pentru a veni în ajutorul roboților cu puteri de calcul și limitări energetice, sunt mutate din Cloud cât mai aproape de Edge. Două obiective principale au fost deja atinse și utilizate ca punct de plecare în modelul propus de noi: servicii de învățare automată în Cloud și un robot cu senzori și motoare. În plus implementăm partea de calcul Fog și extindem modelul orientându-l spre o arhitectură bazată pe microservicii, despre care vom discuta într-o secțiune următoare.

## 6.2 Categorii de informații și analiza performanței

**Modele**. Având un sistem eterogen, datorită diferențelor de capacități hardware ale fiecărui dispozitiv care duce la diferențe în ceea ce privește datele generate, trebuie să definim un mod uniform de structurate a informațiilor. În acest fel, identificăm trei categorii principale:

- Date
- Sarcini
- Resurse

Aceste categorii constituie pachetele care se deplasează în interiorul arhitecturii, între nodurile din toate cele trei nivele (figurile 6.2, 6.3, 6.4).

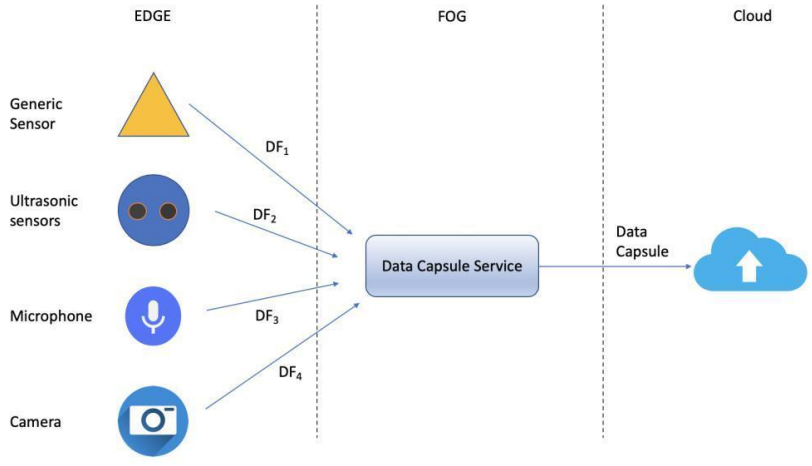


*Figura 6.2. Capsula de Date - Flux de Date.*

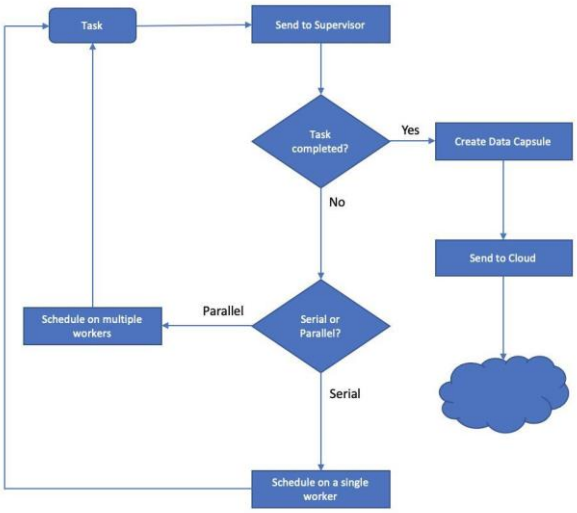


*Figura 6.3. Fluxul de sarcini (bazat pe context).*

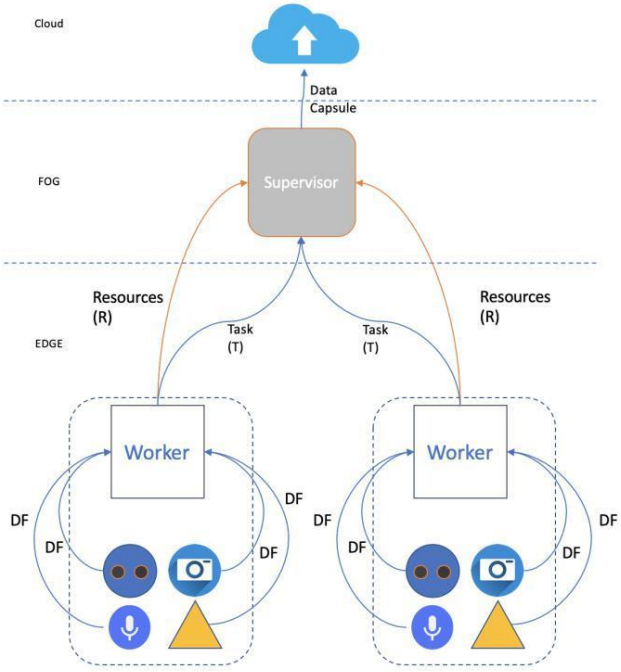


*Figura 6.4. Fluxul de date și sarcini.*

**Analiza performanței planificării sarcinilor.** În primul rând, testăm receptivitatea sistemului. Există multe scenarii în care putem testa acest lucru, cum ar fi: O singură sarcină trimisă unui singur nod worker; Mai multe sarcini trimise continuu unui singur nod worker; Sarcini multiple trimise mai multor noduri worker.

Pentru o singură sarcină, rezultatele sunt prezentate în tabelul 6.1. După cum putem vedea, abordarea monolitică este mai bună dacă există o singură sarcină primită la un moment dat și există suficiente resurse pentru procesarea datelor. Diferența de timp este rezultatul sincronizării mesajelor cu supervizorul și a timpului de colectare a datelor de la fiecare microserviciu. Mai mult analizăm modul în care timpul de răspuns este afectat prin efectuarea mai multor sarcini pe același lucrător, astfel încât procesarea și înregistrarea se suprapun.

Rezultatele sunt prezentate în tabelul 6.2. Aceste rezultate sunt cele pe care le căutăm.

Având mai multe date de procesat și solicitări continue de la utilizator, abordarea monolitică nu poate descărca sarcini unui alt lucrător. Prin urmare, trebuie să proceseze datele în mod solitar. Dificul-tatea este dată de faptul că timpul necesar procesării primelor date este suficient de lung pentru ca următoarea

solicitare să se suprapună cu prima. Prin urmare, va exista o întârziere din cauza acestei suprapuneri în sarcinile de procesare. Acest scenariu nu se poate întâmpla în arhitectura propusă, având în vedere faptul că există suficiente noduri worker care pot prelucra sarcinile prim-ite. Alte avantaje în ceea ce privește utilizarea resurselor sunt discutate în continuare.

| Monolitic | Arhitectura propusă | |
|---|---|---|
| | Planificat pe același nod | Planificat pe nod diferit |
| 20 | 25 | 29 |

**Tabel 6.1.** Comparația timpului de răspuns (secunde) pentru o singură sarcină

| | Monolitic | Arhitectura propusă |
|---|---|---|
| Prima sarcină | 28 | 32 |
| A doua sarcină | 40 | 33 |
| A treia sarcină | 75 | 33 |

**Tabel 6.2.** Comparație timp de răspuns (secunde) pentru mai multe sarcini

În al doilea rând, analizăm diferențele de resurse utilizate pentru procesarea aceluiași număr de sarcini de către ambele arhitecturi. În acest caz, avem în vedere doi roboți și următoarea sarcină de lucru: 20 de sarcini pe robotul 1 și nicio sarcină pe robotul 2. Facem acest test tocmai pentru a vedea dacă echilibrarea sarcinii efectuată prin utilizarea programatorului de sarcini în cadrul supervizorului funcționează cum a fost planificată și, de asemenea, dacă nu utilizează o cantitate importantă de resurse. Rezultatele obținute sunt prezentate în tabelul 6.3.

| Resursă | Monolitic | | Arhitectura propusă | |
|---|---|---|---|---|
| | Worker 1 | Worker 2 | Worker 1 | Worker 2 |
| Baterie (%) | 17.32 | 1.08 | 6.06 | 8.16 |
| Procesor (%) | 72.20 | 4.45 | 13.5 | 51.84 |
| Temperatură (C) | 80.5 | 56.00 | 71.1 | 68.8 |

**Tabel 6.3.** Compararea resurselor

După cum putem vedea, există o diferență demnă de luat în considerare în utilizarea procesorului. În abordarea monolitică, o sarcină mai mare a procesorului pe un singur worker, din cauza imposibilității descărcării sarcinilor pe al doilea worker a rezultat o utilizare slabă a resurselor sistemului. Prin urmare, temperatura plăcii a atins o valoare destul de ridicată și, prin urmare, crește și consumul bateriei. În acest fel, putem observa că arhitectura propusă păstrează încărcarea procesorului și, prin urmare, temperatura în parametrii normali și, de asemenea, obținem o durată de viață mai bună a bateriei. Un alt criteriu important de evaluare este transferul întregului sistem, ca număr de sarcini care pot fi finalizate. Dacă un număr mare de sarcini este primit de un singur worker, în abordarea monolitică, nodul rămâne fără baterie și, prin urmare, disponibilitatea serviciului este afectată, ceea ce afectează și randamentul. În cele din urmă, am monitorizat utilizarea lățimii de bandă a nodurilor worker și supervizor. Rezultatele sunt prezentate în tabelul 6.4.

| | Worker 1 | Worker 2 | Supervisor |
|---|---|---|---|
| Recepționat | 78.80 kBit/s | 109.81 kBit/s | 619.66 kBit/s |
| Transmis | 258.05 kBit/s | 200.54 kBit/s | 608.37 kBit/s |

**Tabel 6.4.** Utilizarea rețelei în timpul descărcării și procesării sarcinilor

Acest lucru ne arată că rețeaua nu este o problemă, atât timp cât numărul corect de noduri worker comunică cu un singur supervizor. De asemenea, dimensiunea datelor este importantă în acest scenariu și, prin urmare, trimiterea datelor care urmează să fie procesate către un dispozitiv care este mai aproape (ținând cont de numărul de hop-uri pe care pachetul trebuie să îl parcurgă, de obicei unul sau două în

acest caz) la dispozitivul final realizează o diferență importantă, în comparație cu trimiterea datelor în Cloud.

În concluzie, arhitectura Edge-Fog-Cloud prezentată în această lucrare este o alternativă bună pentru industria IoT, rezolvând una dintre cele mai mari probleme cu care a început să se confrunte paradigma Cloud computing: întârzierea timpului de răspuns datorită creșterii masive a traficului în rețea pentru comunicarea cu Cloud ca sursă de putere de procesare. De asemenea, folosind această abordare am implementat un algoritm de planificare care are sarcina de a descărca sarcini de la nodurile care au mai puține resurse disponibile. Rezultatul a fost o durată de viață îmbunătățită a bateriei, care este încă una dintre problemele principale ale dispozitivelor IoT. Deși nu o putem rezolva în întregime, contribuim la creșterea eficientei acesteia până la găsirea unei surse de energie mai bune. În acest fel, anumite dispozitive ar putea profita de tehnologiile de producere a energiei, cum ar fi panourile solare, care reprezintă una dintre cele mai utilizate tehnologii ca sursă de energie verde. În plus, modelul propus poate fi îmbunătățit în viitor, atât din punct de vedere hardware, cât și software. Unul dintre avantajele principale ale procesării Edge / Fog este că tehnologiile Cloud de anvergură redusă sunt aproape de dispozitivele finale. Prin urmare, Bluetooth 5.0 poate fi utilizat pentru transmisia de date între dispozitivele IoT și supervizori, deoarece consumul de energie este mult mai mic față de cazul rețelelor wireless. Cu toate acestea, diferența de viteză de transfer de date este de asemenea considerabilă, dar merită testată, deoarece ar putea avea un impact important asupra duratei de viață a bateriei. Mai mult, o actualizare foarte utilă pentru nodurile tip worker ar fi un sistem de poziționare mai bun, combinând datele GPS cu cele furnizate de accelerometru și giroscop. Împreună cu algoritmul simplu de evitare a coliziunii, putem defini un caz de utilizare mai bun pentru descărcarea sarcinilor: un robot care primește o solicitare a utilizatorului pentru a înregistra o întâlnire și nu are suficiente resurse poate trimite sarcina către un alt robot care poate veni dintr-o altă cameră. În cele din urmă, cloud-ul ar putea folosi datele colectate de la nodurile worker ca și Capsule de Date pentru a găsi modele în solicitările utilizatorilor. În acest fel, aceste informații pot fi utilizate de programatorul de sarcini pentru a-și îmbunătăți eficienta, deoarece ar fi conștient de posibilele sarcini în viitor.

# 7. Concluzii

Prezentul raport reprezintă descrierea activităților de cerectare și a rezultatelor obținute de consorțiul proiectului în cadrul etapei a 3-a a proiectului ROBIN. Se poate constata faptul că s-au continuat și exploatat cercetările din etapele anterioare în cadrul fiecărui proiect component și s-au stabilit legături între proiecte. În același timp fiecare proiect component a dat naștere la rezultate, publicații, tehnologii originale, utilizabile în diferite alte contexte și aplicații. Este de asemenea de subliniat faptul că cele 5 proiecte componente reprezintă o viziune unitară a conceptului de roboți autonomi, inteligenți, fie că aceștia sunt utilizați pentrua sistenâă socială sau interacțiune cu clienții, fie că sunt utilizați pentru pilotaj automat.

S-a realizat Workshop ROBIN 3 în Oct. 2020, s-a prezentat lucrarea invitată (key-note) "Social and Assistive Robots Interacting with Humans" în cadrul conferinței ECAI-2020 (12th International Conference on Electronics, Computers and Artificial Intelligence, 25-27 Iunie 2020). S-au publicat în reviste indexate ISI cu factor de impact 9 lucrări (publicate), 2 acceptate (nepublicate încă) și 5 în curs de evaluare, s-au publicat 20 de lucrări în proceedings de conferințe internaționale indexate, au fost publicate 2 cărți în Springer, s-a dezvoltat un produs software robot asistiv (secțiunea 2) și 4 tehnologii noi: platforma robotica (secțiunea 2), asistare șofer (secțiunea 3), dialog în limbaj natural pentru micro-lumea unui robot asistiv 2020 (secțiunea 5), soluții cloud-edge de planificare (secțiunea 6).

Director de proiect
Prof. Adina Magda Florea