UNIVERSITÀ DEGLI STUDI DELL'AQUILA

Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

Università degli Studi dell'Aquila

COGNITIVE ROBOTICS SYSTEMS ENGINEERING

A PILOT PROJECT OF AN EMPATHIC ROBOT

Università degli Studi dell'Aquila - DISIM - Ingegneria Informatica e Automatica

Supervisor: Prof. Giovanni De Gasperis

Co-Supervisor: Prof.ssa Laura Tarantino

Student: Agnese Salutari

WHAT IS BLOCKSBOT?

BlocksBot is a pilot project of an Empathic Robot I developed and I am currently expanding with Professor De Gasperis

- A Robot is Empathic when it can recognize human emotions and properly reacts
  - This reaction can comprise some action and the simulation of an emotion
- Robots have to work with people in more and more contexts, so Empathic Robotics is now spreading fast

For this first experiment, I used Nao 5 robot.

BlocksBot is available on GitHub: https://github.com/agnsal/BlocksBot.

WHY BLOCKSBOT?

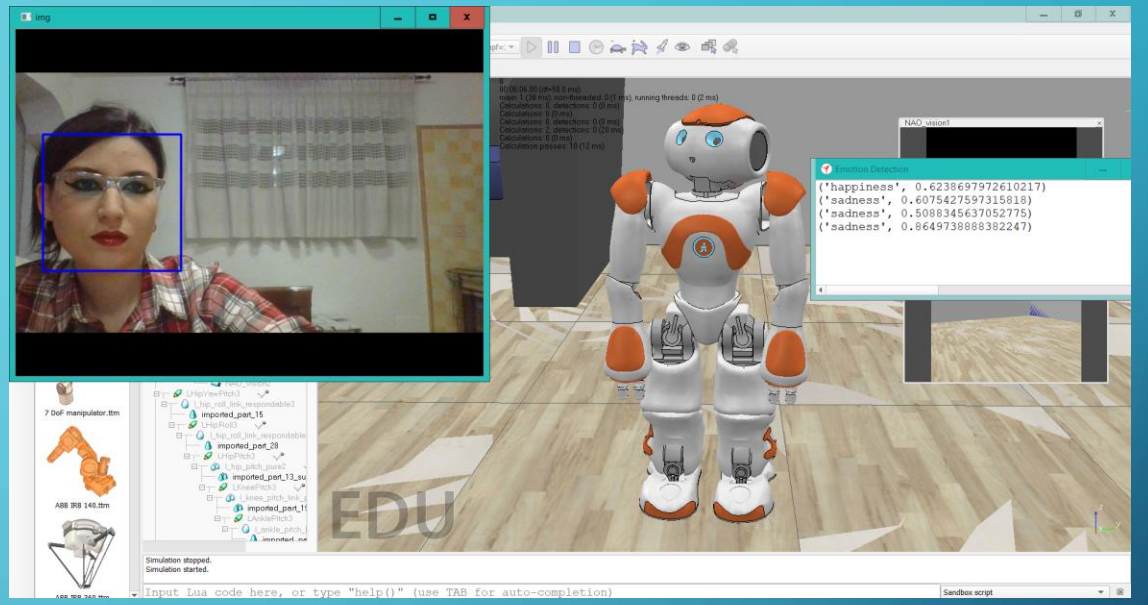
BlocksBot provides common and easy to use tools for building Empathic Robots as hybrid distributed systems in simulations and real world.

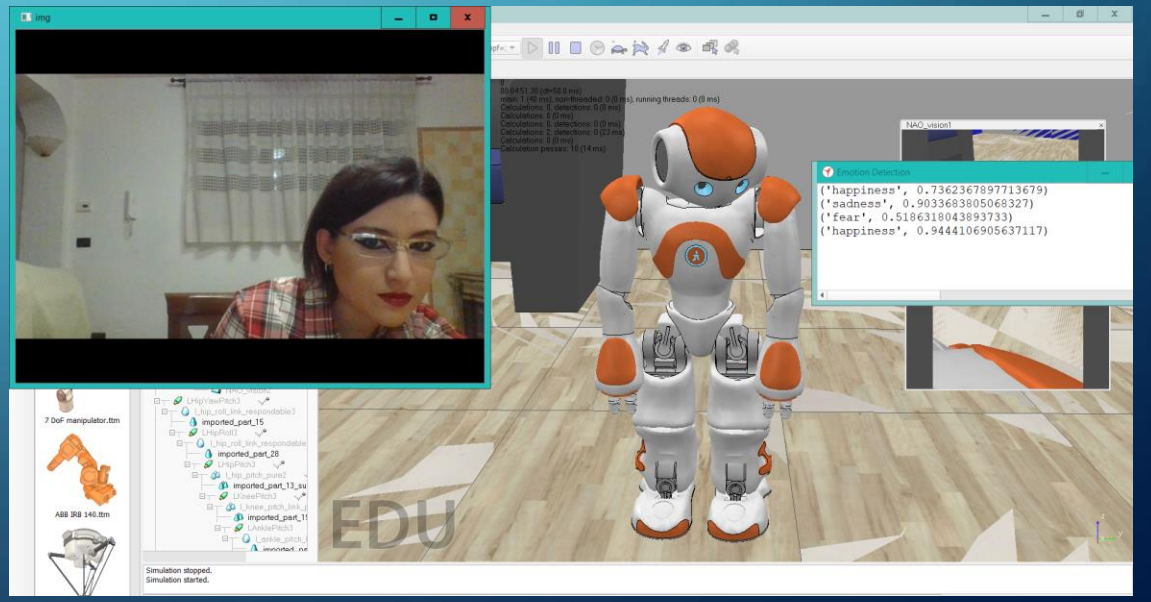
So BlocksBot is

- Portable, in order to save time
  - Simulation code = Real code
  - Develop different robots at the same time
- Modular, developer can quickly plug all needed parts
  - No compatibility issues to solve
- General, to allow high-level development
- Highly compatible, to run on as many kind of Robots as possible

WHAT CAN BLOCKSBOT DO?

- BlocksBot can be used for both simulation robots and real ones
- BlocksBot robots detect emotions looking at
  - User facial expressions
  - User pose
  - User voice
- BlocksBot robots combine these different information to get the most probable emotion and properly react to it while performing other tasks





HYBRID APPROACH FEATURES

Hybrid approach allows to mix different coding tecniques

- Procedural programming
  - Specifying the process step by step
- Logic Programming
  - Describing the start state, the wanted state and the rules that can determine the change between adjacent states
- Machine Learning (for example neural networks)
  - Creating the module and training it

DISTRIBUTED SYSTEM APPROACH FEATURES

Distributed System approach consist of splitting code in many different processes running on different machines

- These processes exchange data via a common broker, based on Redis
  - Processes can publish and subscribe messages on topics, called Redis Channel
  - Redis can be used for data storage too (as a No-SQL Database)
- Each process can be based on a different technology
  - A system of that kind, like BlocksBot, is called Distributed Hybrid System

BlocksBot application is a distributed MAS, that is a set of distributed agents

BLOCKSBOT AGENTS

The following agents can be used both on real robots or other devices, like computer. Their task is extracting emotion from images and audio recordings and combining all the information to decide the most probable emotion

- FacialEmotionsAgent, based on Face++
- PoseEmotionsAgent, based on Face++
- VocalEmotionsAgent, based on Vokaturi
- DecisionMakerAgent, based on my PyDatalog logic

There are two other agents that can be used for CoppeliaSim simulation to easily manage CoppeliaSim robots, to record audio files and images and to perform face detection on images and perform simulation reactions

- AudioSimulationManagerAgent
- VideoSimulationManagerAgent, based on my BlockBot library and OpenCV library

Face++ 旷视 AI Open Platform

VOKATURI

OpenCV

BLOCKSBOT AGENTS

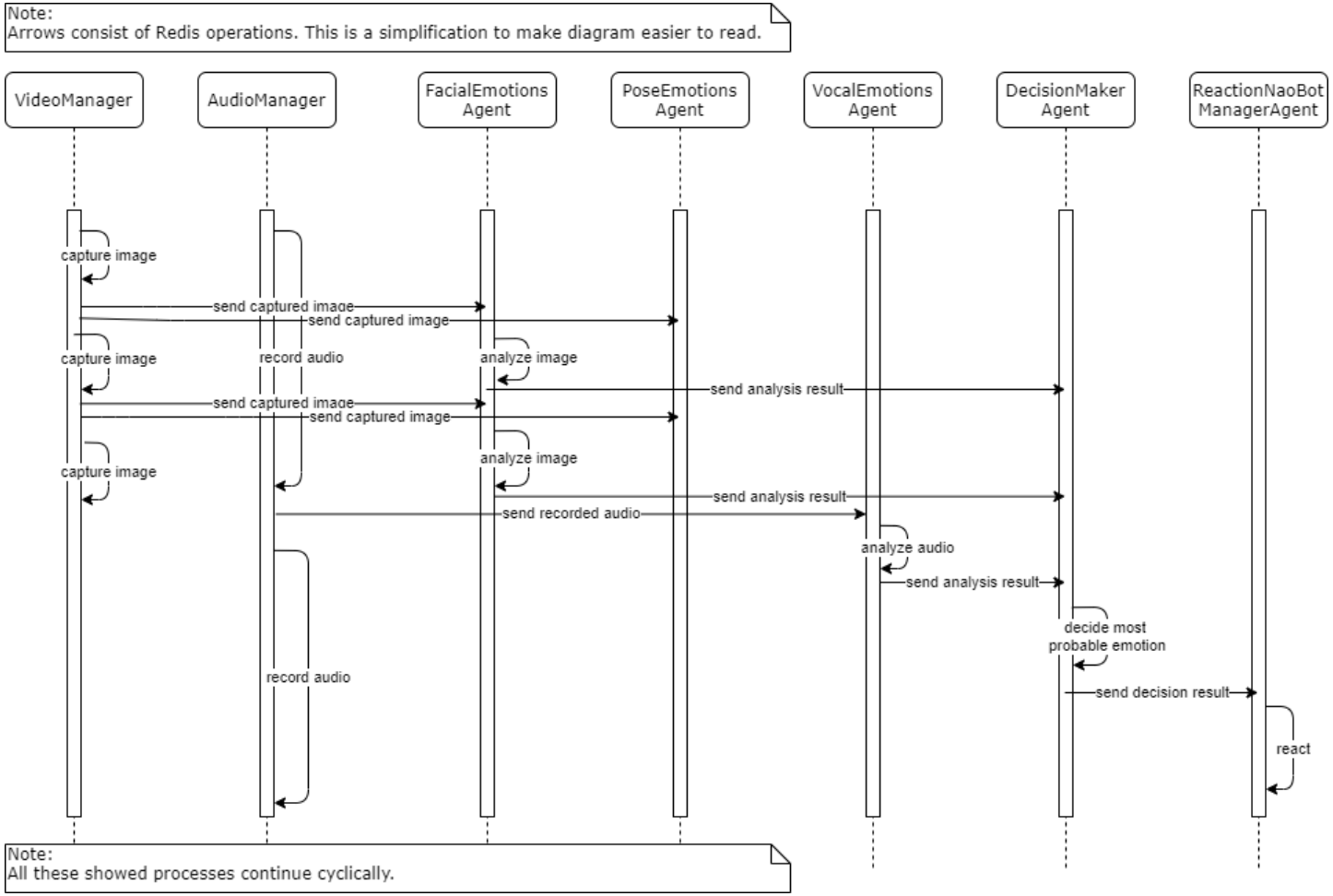
There are three Python 2 agents for real robot, Nao. They have to record audio, capture images and to make robot perform the configured reaction

- AudioNaoBotManager
- VideoNaoBotManager
- ReactionNaoBotManager

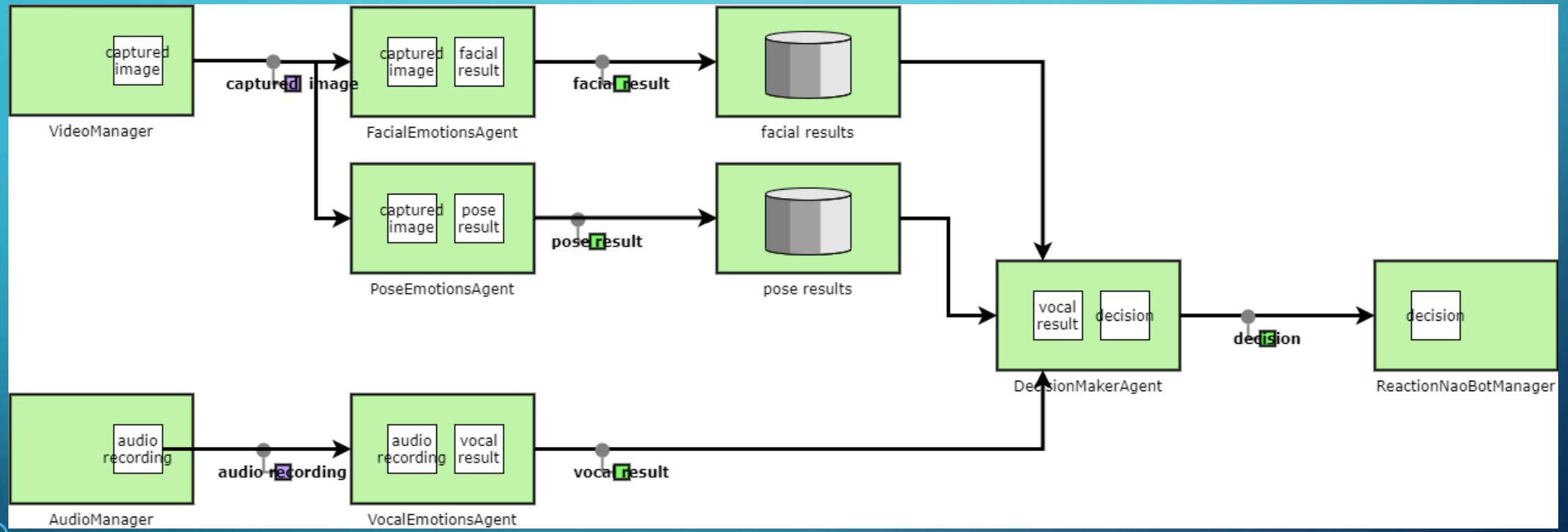
Finally, there are Runners

- Runner is the special agent I developed to start and stop all other agents, in order to have clean start and exit for these parallel daemon processes. It is configurable (you can define which agents it has to start)
- BlockBot provides two Runners to start all other agents
  - A Python 2 Runner for real Nao
  - A Python 3 Runner, that can run on computer

BLOCKSBOT AGENTS



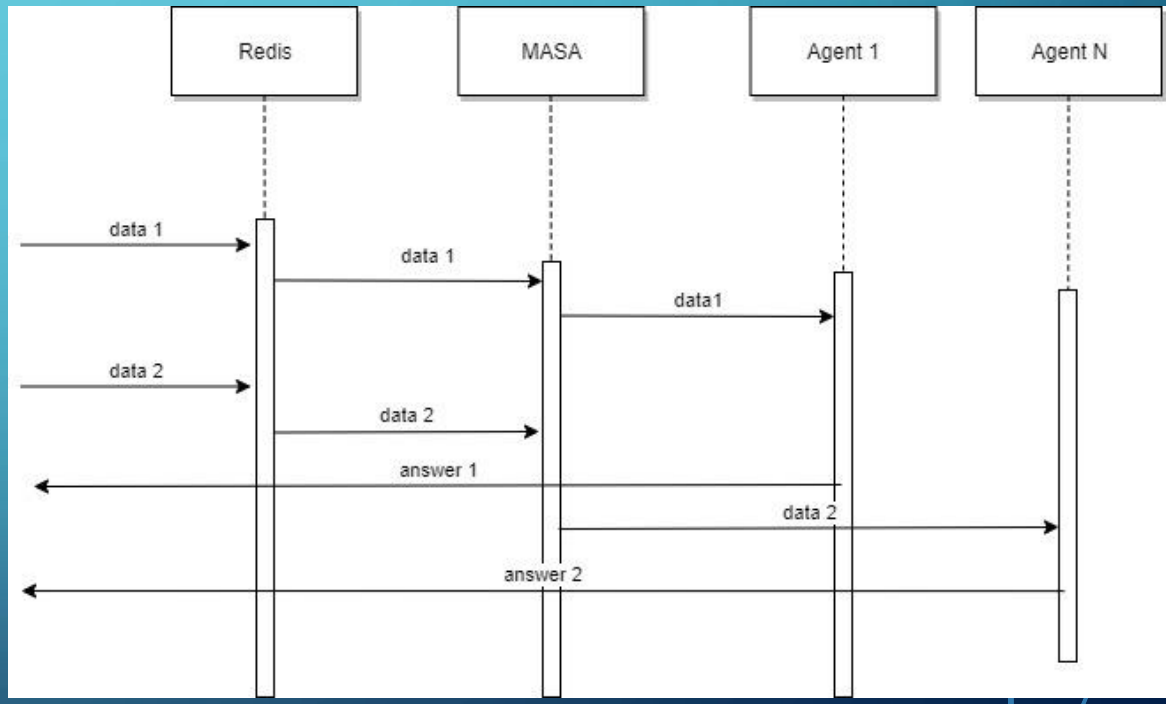
BLOCKSBOT AGENTS



- Facial and pose results are stored in porper Redis queues for a period
- Other messages (captured image, audio recording, vocal result and decision) consist of a notification (Redis publish on a proper channel) and a temporary storage on Redis (for less time than a period)
- Each period ends when vocal result is notified to DecisionMakerAgent and it produces a decision

MY PLUGGABLE ROBOTICS PROJECTS: SERVERDALI

- For my previous thesis I worked on ServerDALI, a MAS - Procedural Programming Hybrid approach. This first system is based on a DALI MAS, integrated in a web server running in a Docker Container: https://github.com/agnsal/docker-ServerDALI
- Then, in Intelligent Systems And Robotics Laboratory course and I improved it, developing more efficient modules for Python 3 applications and making it able to be distributed over many machines. This system is based on different MASs, programmed with different languages, and Object Oriented modules, all connected together via a broker based on Redis. This new version of the system, called KOINE DALI, that is my extension of DALI framework: https://github.com/agnsal/KOINE-DALI

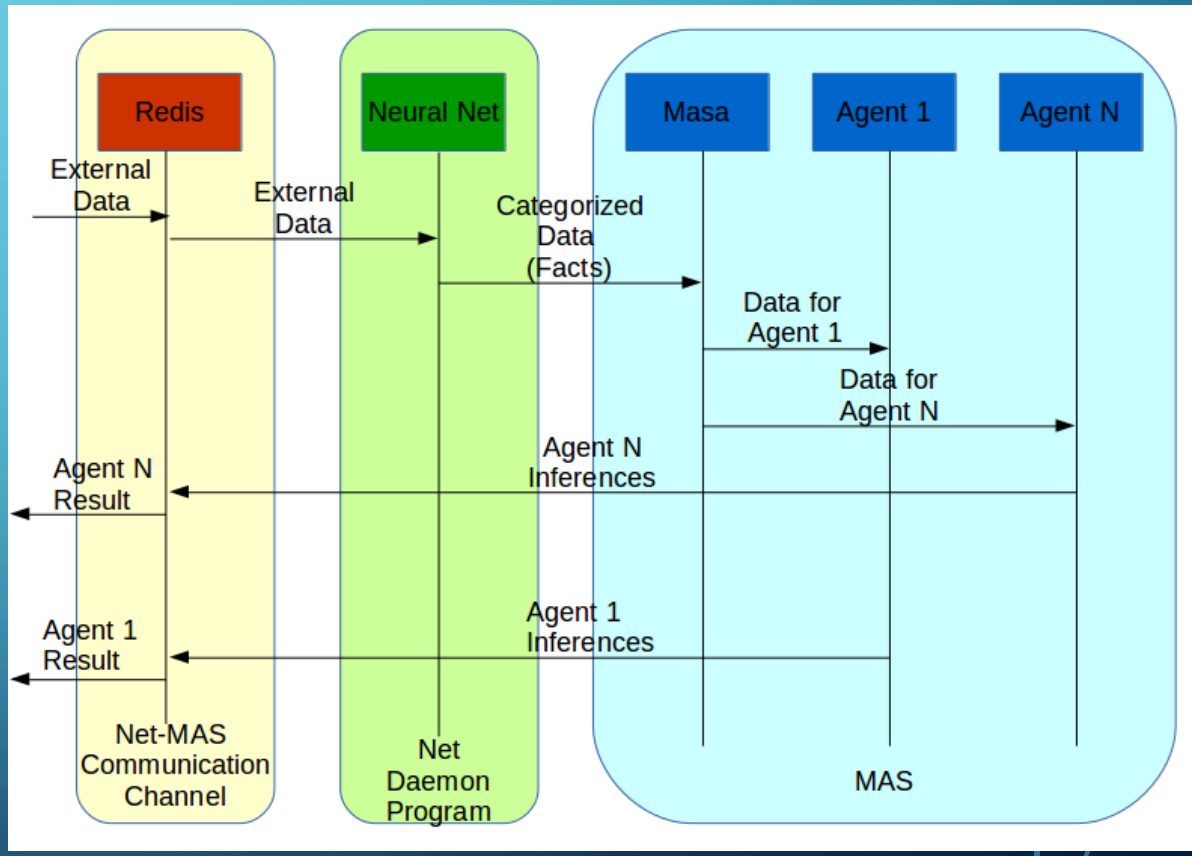


MY PLUGGABLE ROBOTICS PROJECTS: NEURALMAS

Neural Networks are often used for making predictions and classifications over big data. To allow working with both MASs and Neural Nets, during Machine Learning course I worked on NeuralMAS project. This system includes

- A Keras Neural Net
- A Koinè DALI MAS
- A Redis Broker

Code is available at: https://github.com/agnsal/NeuralMAS



MY PLUGGABLE ROBOTICS PROJECTS: CRAZYREDIS

- Drones can guided by other systems or at least send data for these systems, that are maybe hosted by robots or that are hosted by a server called by robots
- During Modelling and Control of Communication Networks course, I developed CrazyRedis, a Python 3 library to easily manage Crazyflie 2 drones API with callbacks and to receive log data via Redis. The project is available on GitHub: https://github.com/agnsal/CrazyRedis

MY PLUGGABLE ROBOTICS PROJECTS: LR2

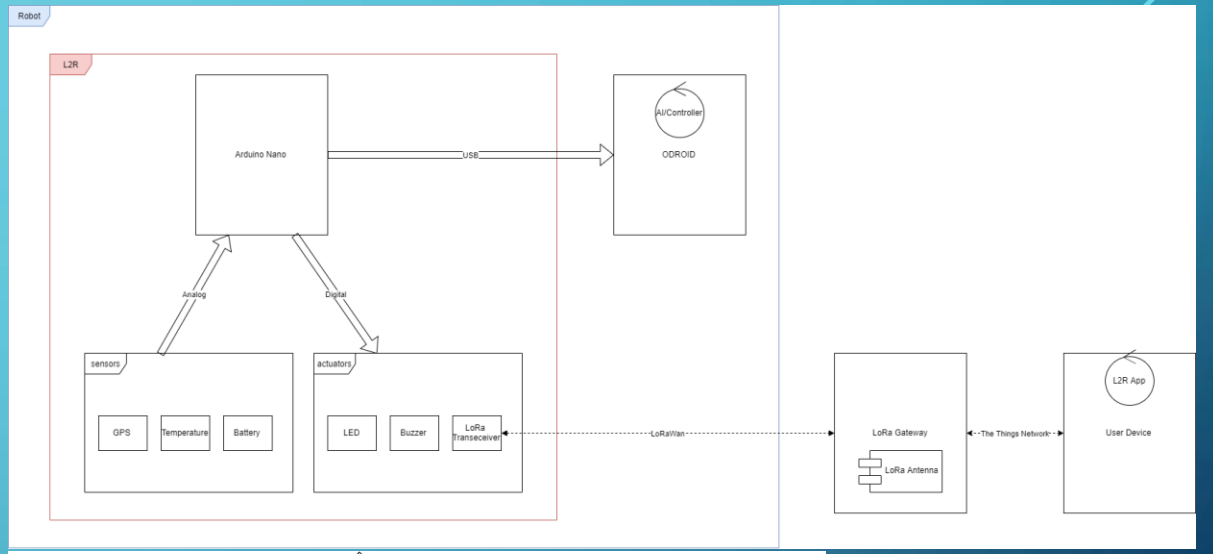
Lora Robot Rescue, or LR2, has the purpose of solving most common and annoying outdoor robot problems

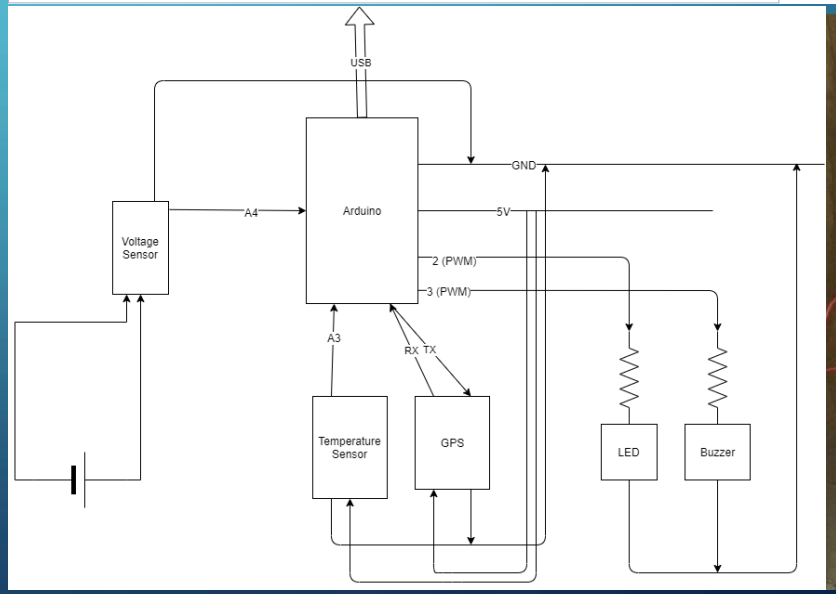
- Limited energy autonomy
- Unpredictable breakdowns
- Troubles with communication in rural areas Furthermore

LR2 is a module that can be installed on a generic Robot to

- Periodically send the GPS location, the charge value of the battery and robot temperature via a LoRaWAN connection
- When, for example, power is under and/or temperature is over a certain threshold, send an alert (containing these parameters) and signal the robot to enter in powersafe mode
- Make the robot more evident, by lighting and emitting sounds, when robot owner is looking for it

LR2 could be a nice antitheft for robots too



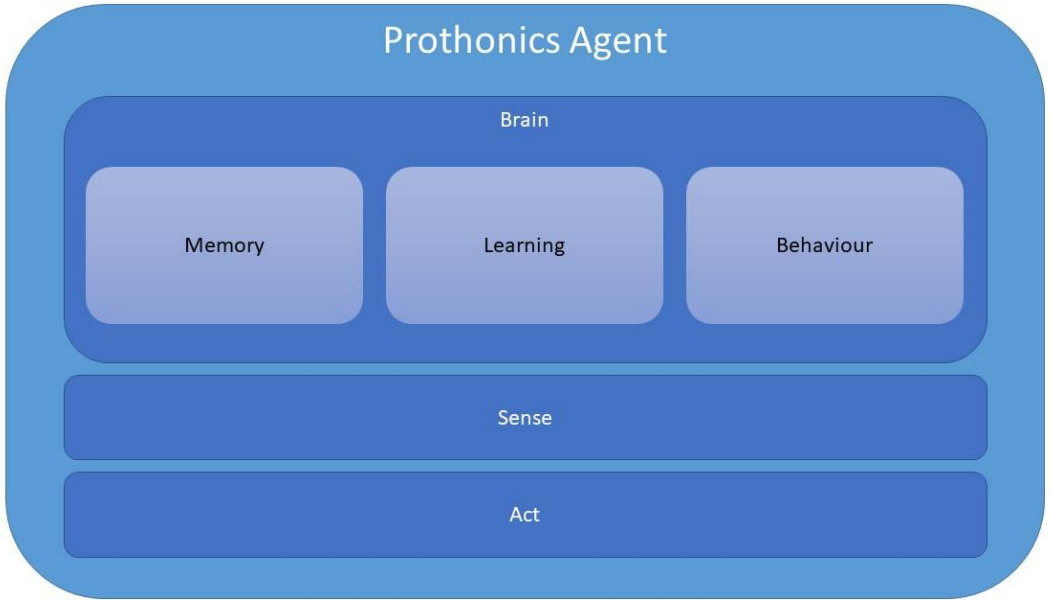


MY PLUGGABLE ROBOTICS PROJECTS: PROTHONICS

I developed Prothonics-vrep and its successor, Prothonics, to allow users to create agents that can perform reasoning using SWI-Prolog in V-REP/CoppeliaSim simulations.

Protonics-vrep and Protonics are both available on GitHub

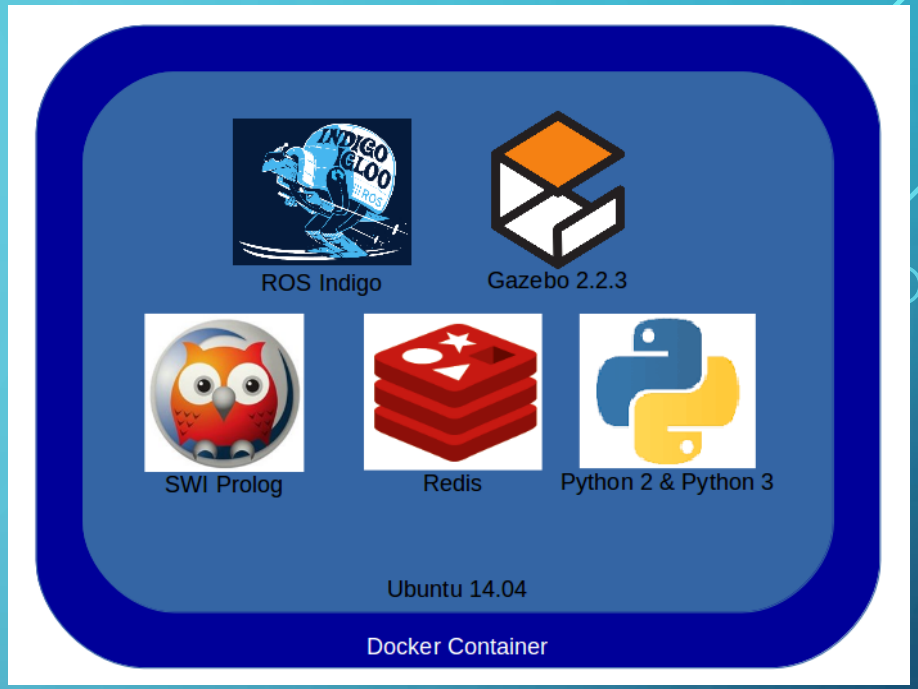
- https://github.com/agnsal/prothonics-vrep
- https://github.com/agnsal/prothonics

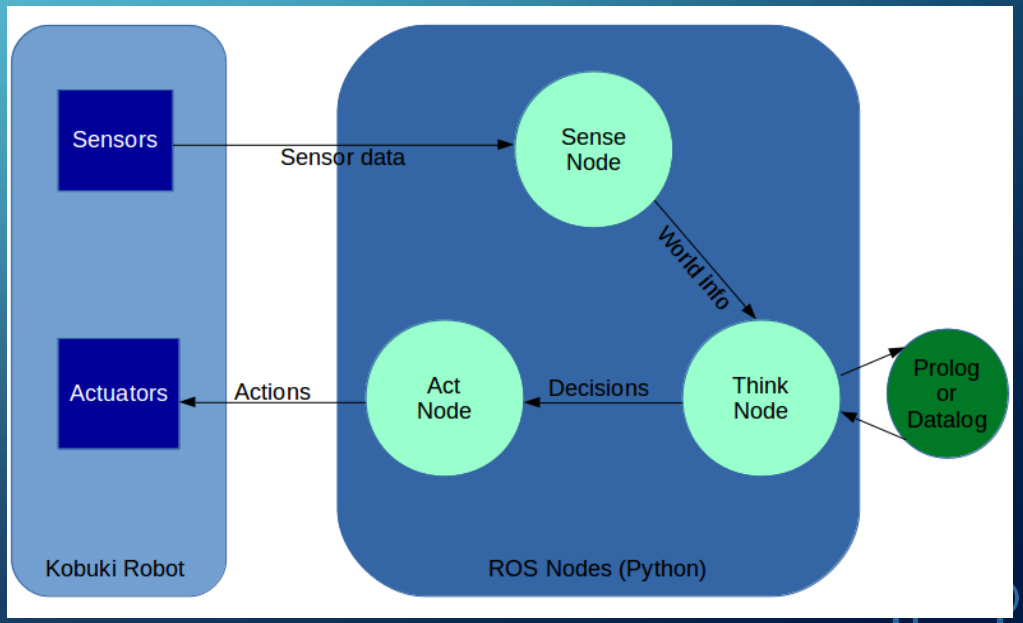


MY PLUGGABLE ROBOTICS PROJECTS: DOCKER-INDIGOROSDISPYPL

To create hybrid applications in a ROS environment, I developed the following modules

- Docker-IndigoROSdisPyPl, a Docker container with all the needed toold to create Python, Python/SICStus-Prolog, Python/SWI-Prolog and Python/Datalog agents in a ROS environment (https://github.com/agnsal/docker-IndigoROSdisPyPl)
- KobukiROSindigo, an example of such a system (https://github.com/agnsal/kobukiROSindigo)





ACKNOWLEDGEMENTS

- Thanks to Università degli Studi dell'Aquila Professors and Staff for having helped me to grow professionally and as a person and for their diligence in these difficult days
- I'd especially like to thank Professor De Gasperis and Professor Tarantino for their support and patience
- Thanks to my parents and family for their closeness and encouragement
- Finally, thank you for your attention