Interdisciplinary course of

Design and Robotics

10° edition, 2022

Project:
P'Hug
The haptic Robot

Professors:
Andrea Bonarini, DEIB department, Politecnico di Milano
Maximiliano Romero , Design department, IUAV, Venice University

Tutors:
Federico Espositi

Students:
Alessandro Barbiero, school of Engineering
Elias Insulander, school of Engineering
Helen Berhanu Tekle, school of Design
Lorenzo Poretti, school of Engineering
Lorenzo Dondi, school of Design
Luigi Altamura, school of Engineering

Index

Abstract

According to surveys and research, the haptic sense is a significant aspect of human wellbeing. The experience received as a result of contact with another living creature like hugs stimulates serotonin production. High serotonin levels improve mood, make people feel better, and help having better sleep. Nowadays people are more linked than ever before as a result of the developing trend of technology. While technology has helped to bridge the gap, it has not been able to eliminate all the barriers imposed by distance. Our initiative primarily aims to bridge this gap by creating a product that sends a haptic hug from one device (Giffy: the puppet) to another one (the Jacket). The embrace and caress from the puppet is transferred via a wifi connection to the wearer of the jacket. The wearer can feel three major sensations: vibration achieved by vibration motors simulating a moving caress, pressure from an inflatable chamber simulating a hug, and a constant warm feeling from a resistive wire. Our goal is not to replace an actual embrace, but to help people that live apart and cannot hold their loved ones having a fraction of that desired contact.

Phase 1: Discover

At the beginning of this journey we tried to understand each other and how every component of the group could be useful for the final goal. So first we got a brief description of our area of study and personal interests and then we decided on the tools for the communication. We mainly used Telegram to share files and ideas, as well as schedule our meetings, whereas the online calls were handled via Discord. We tried to meet in presence as much as possible given the physical nature of the project, but the online communication was a crucial part too.

1.1 - Team Organisation

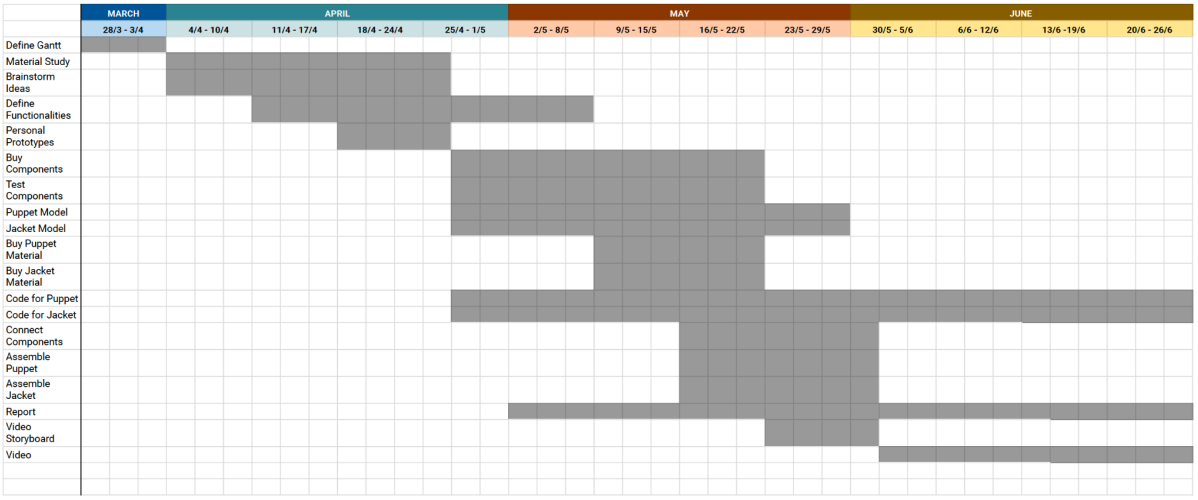
The P'HUG team / team 7 comprises six members:
- Lorenzo Poretti, Luigi Altamura, Elias Insulander, Alessandro Barbiero ➔ the engineers;
- Helen B. Tekle and Lorenzo Dondi ➔ the designers.

The team is led by Luigi Altamura, an engineer with great enthusiasm and positivity and the one with more electronics experience. These reasons and the fact that we use his house to work on our project moves us to choose him as our leader.

Team P'Hug members working

1.2 - Project Management

Our team plans to solve the tasks according to the following Gantt Chart group 7 for the main objectives:



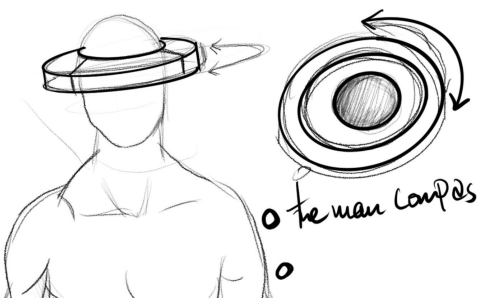
And using Trello.com website functionalities to handle the subtasks.

1.3 - Research

First our aim was to meet up the standards and requirements set by our instructors: Design a wearable robot that interacts with a third person. It led us to many varying ideas to reach our specific goal. We made different navigations around the internet about wearable robots and interactions in order to get the best outcome and put all the results on a shared board on Miro.com.

Step by step from emotion detectors to haptic technology

Our first thought was to create a robot placed on the shoulder that recognizes objects at distance and perform different actions based on that. We then switched to a robot able to detect another person's emotion and communicate accordingly, such as by offering tissues, making them laugh, and so on.
Trash eater, Barman aiding robot, and Hat robot are a few examples. For a long time we brought these ideas on in parallel, developing the various possible interactions with all the pros and cons for each one of these ideas.



Our intention was to do something a bit different when we shifted from these first ideas to our next, even though it was a little bit challenging.

To accomplish our goal, we sought to examine the various features of a physical hug and their significance to human physiology and psychology. We gathered as many case studies as we could as input for our work.

Haptic senses

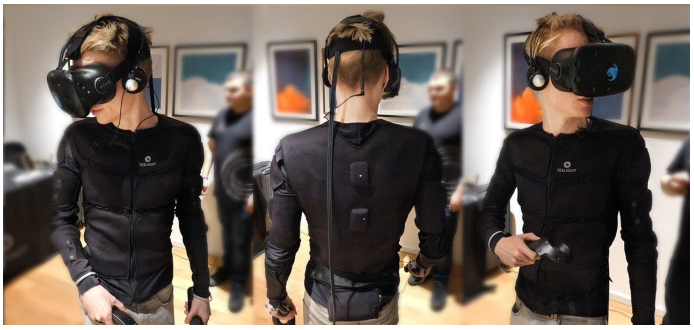
We begin with the general definition of Haptic technology It typically uses an actuator to convert electrical, hydraulic, or pneumatic energy into vibrations, which are then managed and controlled by software that defines the duration, frequency, and amplitude.

Wearable technologies are one of the several innovations for people to be linked. One of our earliest inspirations is the Pillow talk, which combines a long-distance heartbeat monitor with a speaker that slips under one's pillows so that they can hear each other's heartbeat while sleeping. Pillow talk can only send real-time heartbeats if both users wear the bracelet to bed.

Bond Touch wristbands are another technology that use haptic technology to vibrate and light up with a personalised colour when one of you taps on the bracelet twice, similar to how Apple Watches do.

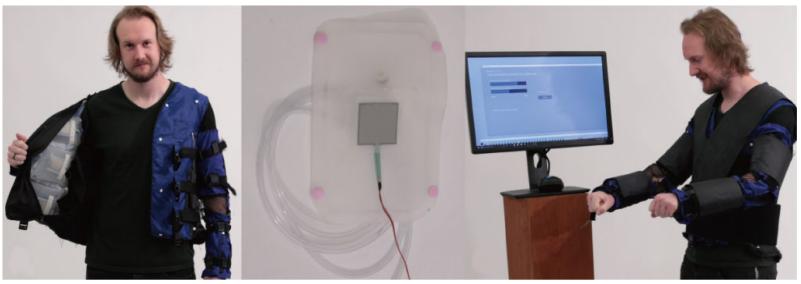
Haptic sensing in the gaming field

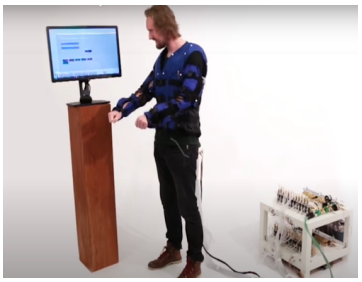
These are intended to help people navigate the full gaming experiences not only by seeing but also from full-body vibrations. A more natural way of interaction passes through a haptic sense.



The Disney Force Jacket

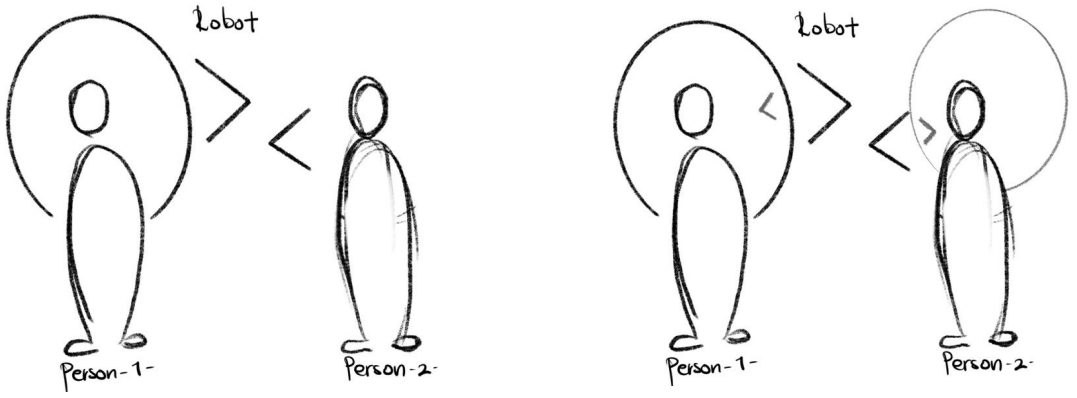
The jacket is designed to be used in conjunction with a virtual reality headset to offer a fully immersive experience. The jacket could be used to replicate actions in the virtual world in the real world. According to different studies over the previous few decades, haptic feedback has played an important role in the entertainment system. So we took a great deal of inspiration from it.



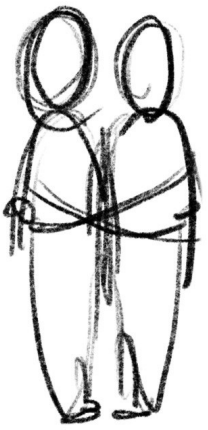


1.3.1 - Robot Strategies

At this point, our overall plan was to execute and understand the instruction given by our instructors and tie it to the research we conducted in the best way possible. We drew inspiration for the case studies from the Disney force jacket, which has an inflatable chamber for comfort, and from haptic technologies such as gaming, which aims to transfer touch feelings from a sending person to a receiving person. With the assistance of our teachers, we transitioned from a less interactive self embrace to two-way interaction.



From a robot that communicates just with a second person to robots that communicate with each other and convey varying sensations.



1.3.2 - Functionalities

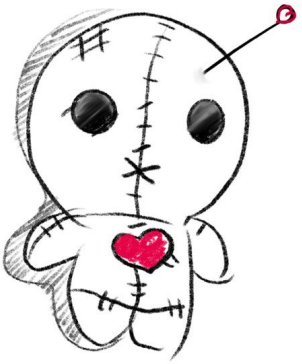
We started from the basics…

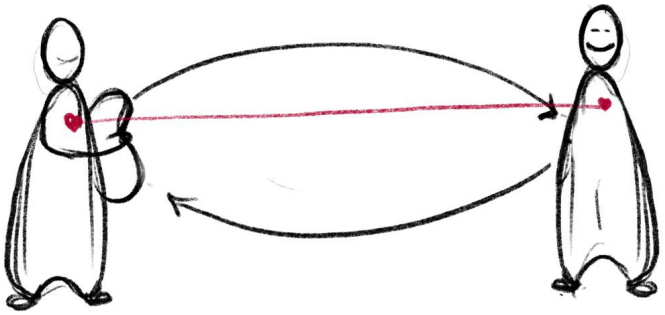
- Where do we embrace people the most?
- Where do we have the most senses when hugged?
- What do we feel when we get hugged?

After identifying them, we proceeded to explore ways to transfer the interactions to another product. This is where the internet comes into play. The key features were characterised as a product that transfers the sensation of a hug from one person to another.

…Like a voodoo doll!

Person 1 touches the product and the other person feels it. Which was more like a voodoo doll which means “things which do things”.

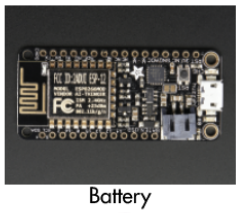




1.3.3 - Materials

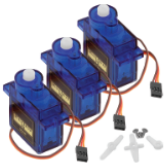
After clarifying the functionalities we roughly discussed the materials and technology we intend to use to realise these functionalities. This was the first draw of the final components we used:

Microcontroller (Adafruit Feather HUZZAH with ESP8266)



Big pressure sensors

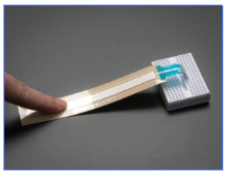
Servo motors



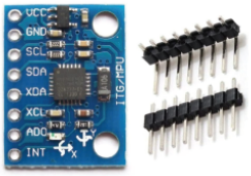
Arduino MKR1000 Wifi

Battery

Pressure sensors for caresses



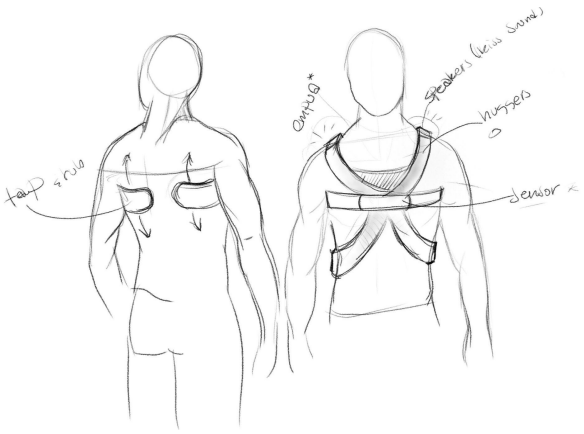
Accelerometer



Vibration motor

Phase 2: Define & Develop

We came a long way to get at the concept of P'hug, from a pet on shoulders to emotion detectors and headgear. After a long discussion, we decided that it would be best to develop the function of the hug enhancer. Hence we decided to study and come up with our own interpretations of the enhanced hug. The hug enhancer operates on the basis of a pose sensor positioned mainly on the wearer's vest around the chest area.



2.1 - First Prototypes

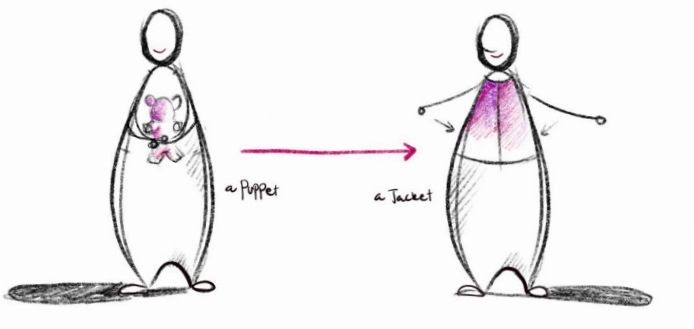
On the 26th of April we all came up with our own interpretation of the self embrace. We learned that the enhanced embrace has very minimal interaction after a lengthy conversation with our professors. Hence we shifted the enhanced hug to a two-way interaction which is sending the hug from one product to another via a wireless network. We were having some difficulty catching up at this point because we had revised the original concept to a wirelessly connected haptic sensation.



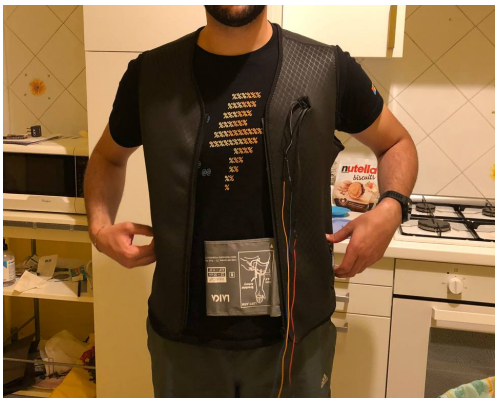



2.2 - Strategy

After April 26th, we decided how we would express the hug. The idea was to have two objects that belonged to two different people separated by a considerable distance. A wireless network connects each object, allowing them to converse and convey touch sensations to users. Following our discussion, we decided to pursue a Jacket and a puppet as preferable products.



In this phase we transformed from the simple prototype phase to something close to our final results. First, we added materials to our list to develop our models. We 3D printed the head for Giffy and put it inside the body, we tried if the ears respond to the caress, we tried if the jacket responds to the hug and we put the air pump in the Jacket.

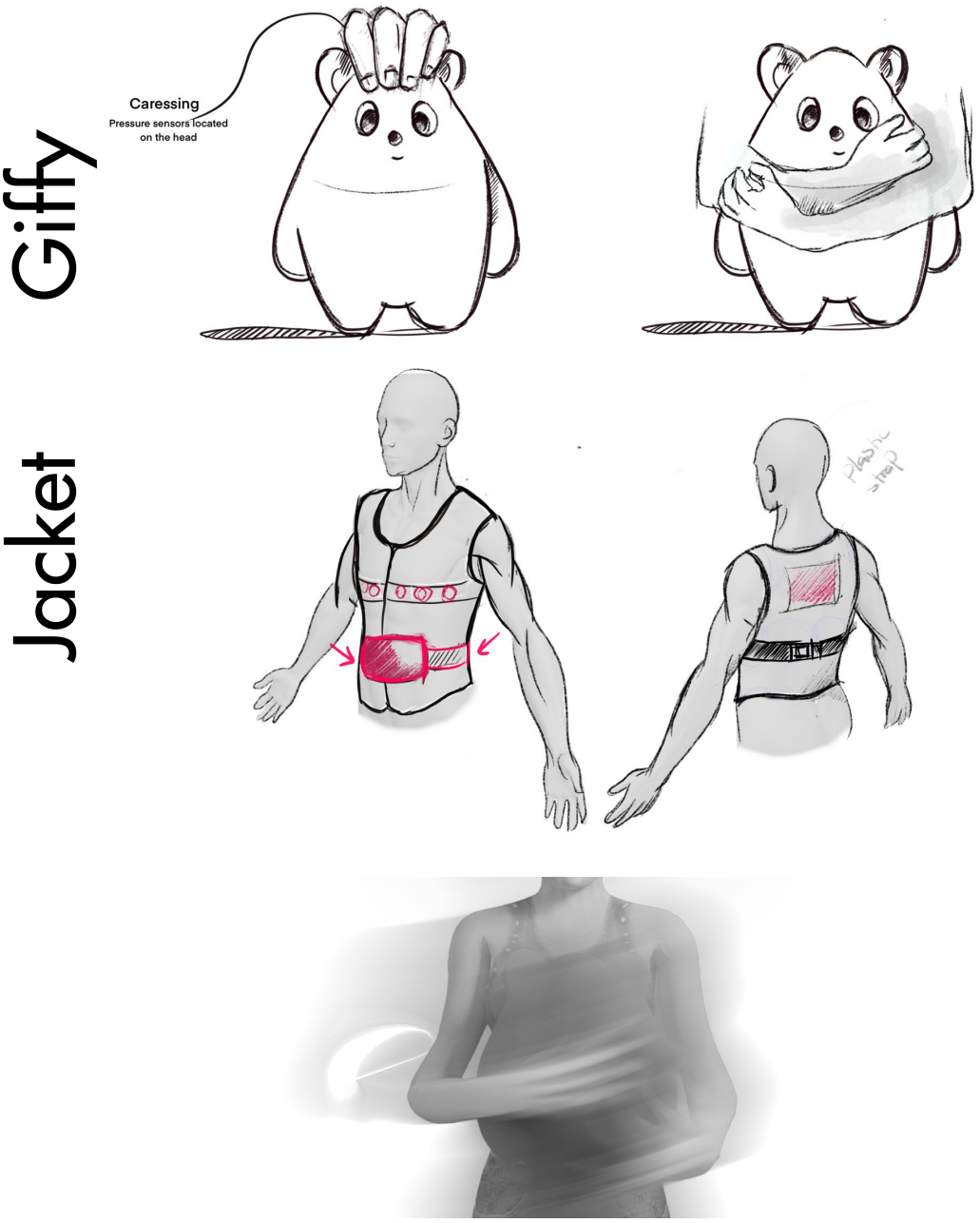


2.3 - Functionalities

Caressing & hugging

The next step was to define the main functions and the consecutive responses.

- Person 1 caresses Giffy on the head: Jacket vibrates, Giffy makes a sound, Giffy lowers its ears alternately
- Person 1 hugs Giffy: Jacket squeezes person 2 on the west, Jaket heats, Giffy moves ear.
- Person 1 shakes Giffy: Jacket vibrates, Giffy shouts



2.4 - Electronics

The most difficult part from the hardware point of view was to apply theoretical knowledge in a real-world setting. It was not easy to understand at first what the difficulties might be, but when we started choosing components we realised that a good practical knowledge was needed. All this led to several problems, but thanks to the experience we gained and a few burned-out sensors, we were able to solve them.

2.4.1 - Giffy

This part of the robot consists of 2 servo motors, 4 force sensors, three for stroking and one for hugging, a speaker, a microcontroller, a switch, a voltage regulator and a battery pack.

Fixing the wires with the microcontroller

Battery pack

To power Giffy we used a battery pack consisting of 4 AA batteries for a total of 6V. However, the batteries in a real setting when fully charged turn out to be more than 1.5V each, especially if they are alkaline batteries. For the project we used a microcontroller that needs to be powered at a maximum of 5 V, and for this we used a voltage regulator.

Voltage regulator

Batteries during their charge-discharge cycle have a voltage that changes over time. Some components need a certain operating voltage, which is why we used a step-down voltage regulator. Therefore, we adjusted a voltage of about 6V to 5V.

Force sensors

Force sensors sense the bending and pressure that is made on the sensor by measuring the resistance that is created based on the intensity of contact.
The biggest challenge was figuring out how to read the resistance from a 3.3V board when the sensors were powered at 5V. Unfortunately, there were not many schematics for lower voltage boards, and we adjusted the resistors in order to have significant input data.
Unfortunately, during the experiment one sensor burned out due to an error in the choice of resistor and was replaced with 3 others of a different type in order to detect the caress.
In order to avoid overloading the hugging sensor we used two resistors one at the power supply and one at the readout in this way we had clearer data and the sensor did not burn out.
One of the biggest challenges has been finding the right connectors for a solution that is solid but at the same time able to allow sensor replacement. After trying to 3D print connectors without success, we modified existing connectors and achieved the goal.

Servo motors

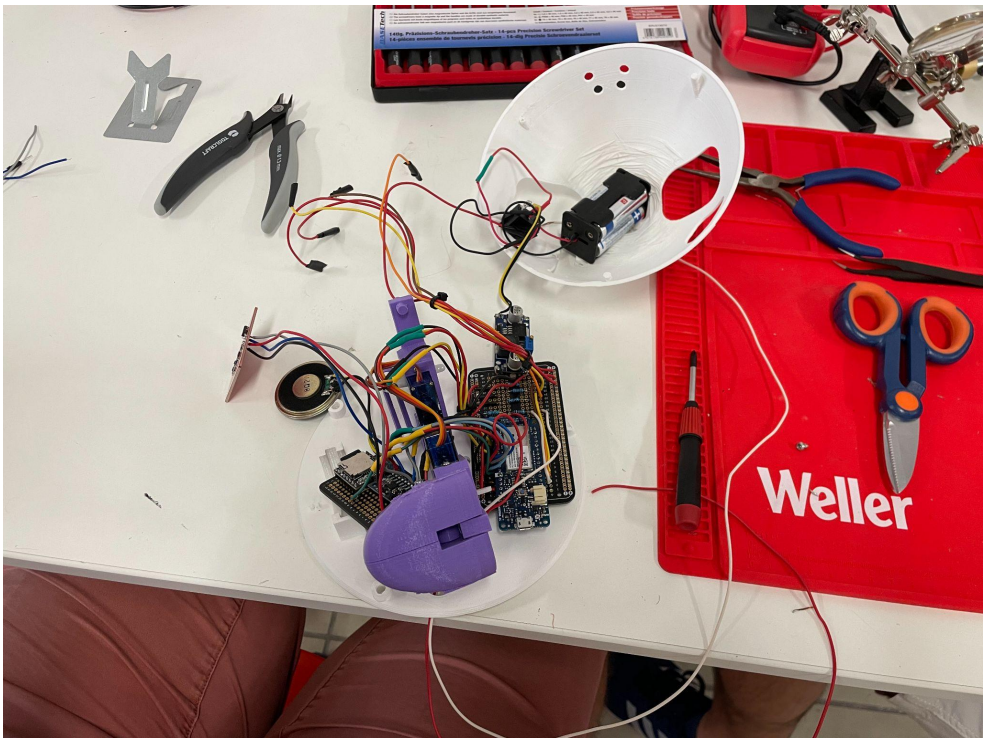
The servo motors were also easy to operate because of our prior knowledge. The only problem was only powering them at 5V since the board runs at 3.3V. We decided to connect them directly to 5V from the voltage regulator.

Speaker

The speaker was also powered at 5V but can be driven at 3.3V. The only difficulty we encountered was the size of the micro SD. The instructions for the component did not mention the maximum capacity it could support. After some trial and error and research on the Internet we found that it supported up to 23 Gb and we were using a 32 Gb one.
We solved it by buying a microSD of lower capacity.

Cable management

On the hardware side, one of the most difficult parts of the Giffy was organising the arrangement of components in the head and finding the right size of cables. Once the near-final version of the head was printed all the cables were carefully shortened and tinned to take up as little space as possible. All this also allows maintenance to be performed in a circuit that is more tidy and clear to understand.



Putting the electronics in the head of Giffy

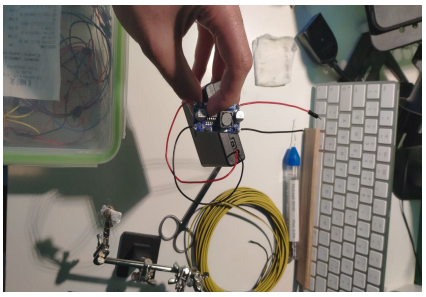
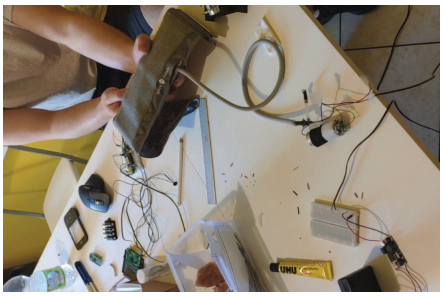
2.4.2 - Jacket

This part of the robot consists mainly of 2 battery packs, 6 vibration motors, one air pump, an inflatable chamber, a relay, two microcontrollers, a switch, two voltage regulators, a driver, a transistor and the heater plates inside the jacket.

Air pump, Inflatable chamber and driver

The air pump and inflatable chamber were taken from a sphygmomanometer.
Learning how the automatic sphygmomanometer works and attaching it to a power supply and a voltage regulator was the most interesting and challenging phase. Furthermore, it has to be linked to the pressure sensors inside the Giffy so that it can inflate whenever pressure is applied.

An inflatable chamber is detached from the Sphygmomanometer with its air pump and connected with a power supply, voltage regulator, and Microcontroller to make some tests. Thanks to a tester we discover the right voltage e and we understand how it works.

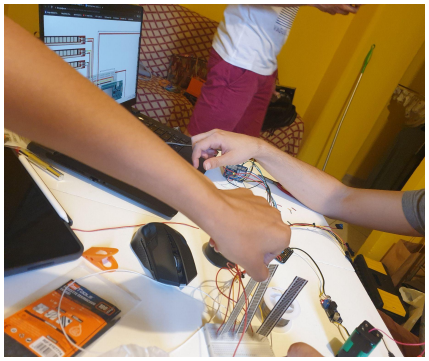
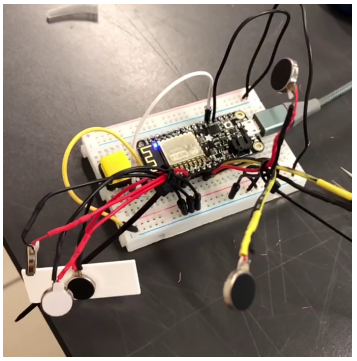



Voltage regulator and power supply

In the final configuration to control the air pump, we also used a driver to handle the return currents that a DC motor can send to the microcontroller.

Vibration motors

The aim of the vibration motors is to convey a caressing sensation that is sent from the puppet to the jacket. Our first attempt to send caresses to the vibration motors was successful.


The main issue with the motors was to manage the maximum amperage that the microcontroller could give to the various actuators. Each vibration motor at 3.3V requires 60 mA for operation at maximum power. We found that 6 vibration motors was the maximum number that could be used, so we had to use another controller for the remaining components of the jacket.

Relay, transistor and heater plates

To be able to control the resistance of the jacket, we used a relay. The problem is that our board can supply a maximum of 3.3V on the output pins, and the relay operates at 5V. To increase the output voltage we used a transistor that we used to amplify the voltage. This was not an easy operation because we had to change the position of the relay ground so that the right voltage difference could be registered. Wiring diagrams will be available in later chapters.

Battery pack and voltage regulator

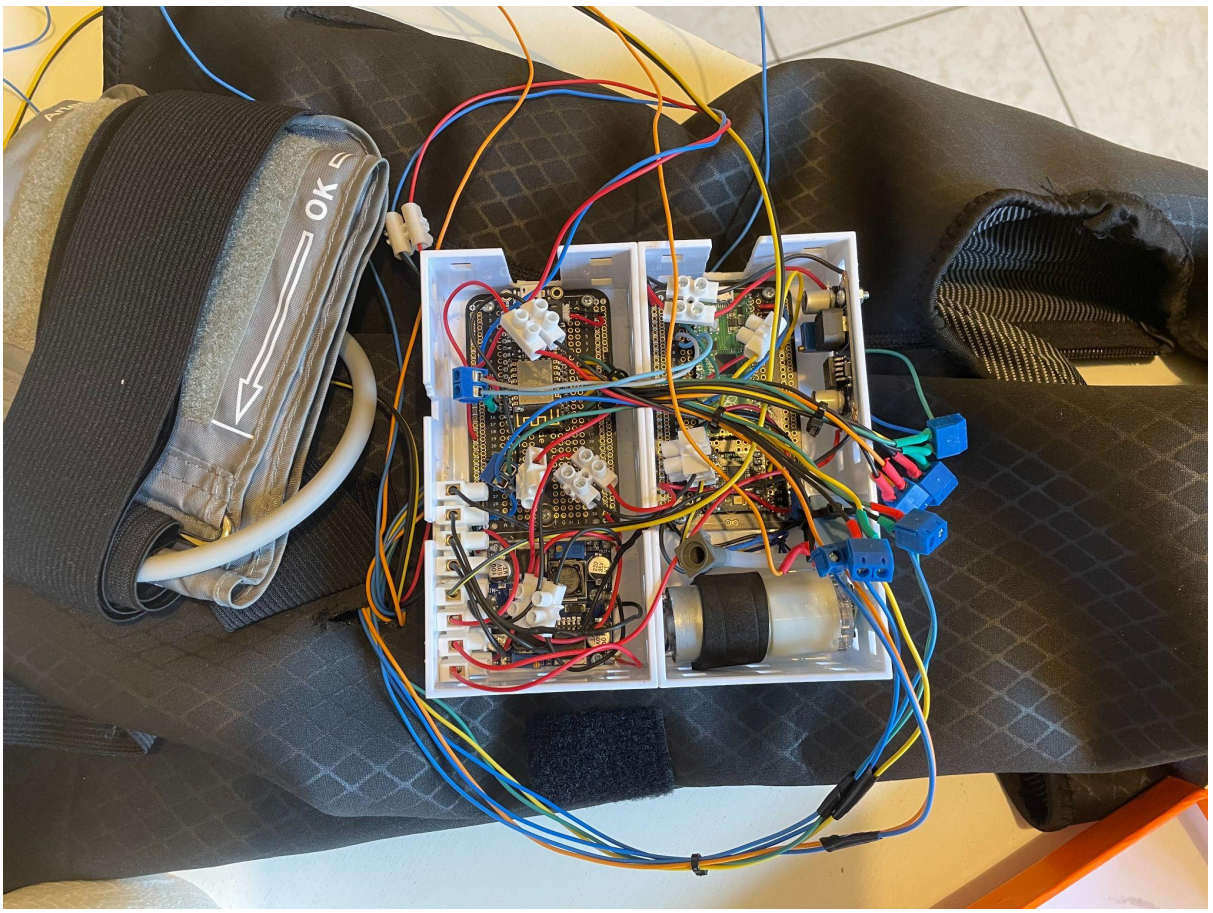
The jacket components use different voltages, so we have 3 voltage regulators that modulate the voltage according to our needs. For example, the air pump uses 6V while one microcontroller uses 5V and the other 3.7.
We decided to use two battery packs, one for all the electronics and one for the resistors that heat the jacket both for a power consumption reason and to give the user the ability to use the jacket without it getting hot during hot weather. For example, in summer, not everyone likes to feel the jacket getting warm.

Microcontrollers

Initially, the jacket was designed to be controlled by a single device. But as the project evolved, we realised that one microcontroller was no longer enough. So we decided to add another one and made them communicate using serial communication. We kept the logic in one controller and the other one performs a slave function by parsing the messages and executing the instructions received.

Cable management

In the jacket, it was essential to arrange the position and length of the wires. We tried to make everything as maintainable as possible by using connectors that would allow the main sensors and actuators to be replaced and also make the boards independent in case of failure.
Another challenge was to organise all the wires so that they would pass inside the jacket without compromising the comfort of use.



Electronics box for jacket

2.5 - Coding

The coding activity has been made by the four engineers together with the help of Github. We started trying to handle the complexity of the project with a simple .ino file on the Arduino IDE but we soon changed to Object Oriented programming utilising PlatformIO.

2.5.1 - Giffy

We knew we had to establish a connection between Giffy and the jacket, and we discussed using Bluetooth or Wi-Fi. When finally deciding to use Wi-Fi, we found useful libraries for both Arduino and Adafruit boards. These libraries helped us with the setup of the server-client communication.

Multithreading issue

Arduino MKR1000 and Adafruit Huzzah 8266 microcontrollers do not support multithreading. We decided to use the millis() function of the Arduino.h library in order to build our custom made timers and simulate a multithreading behaviour.

Accelerometer issue

One of the main issues was reading the accelerometer values. In particular the program used to crash without any clear reason when the speaker and the accelerometer tried to communicate in parallel.
Initially we thought the problem was inside the speaker class but at the end we figured out that the problem was instead the accelerometer class.
After several attempts to understand the reason we finally came to a possible conclusion: Apparently it was not our problem but a library communication problem, there was a conflict in the parallel data transmission between the speaker and the accelerometer when it was trying to read IMU-related values. Most likely when the audio was on, the accelerometer returned a null pointer and this caused a stop in the application.
After many attempts to modify the code and also the underlying library, we decided to move from Adafruit_MPU6050.h library to MPU6050_tockn.h library.
After slightly modifying the accelerometer class, the new code worked and the program did not crash again.

Ear issue

In order to move the ears alternately during hugs, we implement a sequential code which makes the ears seem to move in a parallel way. We use a discrete time model to reach it.
At each time instant the HugHandler calls the idle function inside the Ear class.
This function checks the ear state and based on this indicates the next angle the servos have to reach.

NFC issue

PN532.h is a high level library which is most used by the community to read and write NFC tags. Unfortunately it was not possible to compile the code for the Arduino MKR1000 board while it works for other boards like Arduino Uno.
We decided to use
We decided to use Adafruit_PN532.h as an alternative library which, however, is a lower level library with respect to the PN532 library.
In order to understand how the memory of a Mifare Classig card works we used this manual https://shop.sonmicro.com/Downloads/MIFARECLASSIC-UM.pdf.
We didn't find helpful documentation to read and parse the NFC payload so we implemented such a function.

2.5.2 - Jacket

During our first tests we created small programs to figure out how to use the different components, such as vibration motors. Then we switched to a more elegant and functional DEBUG mode inside our main code.

```
void loop() {

  #ifdef DEBUG
  exTime = millis();
  caressUnitLeft.run(exTime);
  caressUnitRight.run(exTime);
  hugUnit.run(exTime);
  heatUnit.run(exTime);
  delay(3);

  if(exTime - startTime > 5100){
      caressUnitLeft.start(exTime, INTERVAL, SHIFT);
      caressUnitRight.start(exTime, INTERVAL, SHIFT);
    startTime = exTime;
  }
  #else
  //Real code
```

Disconnect issue

One issue we had during software development occurred for the disconnection of the client and server. When the client was switched off, the connection was still active and the client was not disconnected automatically like we hoped. We therefore decided to create a custom pinging system. This means the client continuously sends pings at a set interval and if the server does not receive a ping for a certain amount of time, the server forces a disconnect.

Parsing serial communication

The messages sent from the Adafruit board to the Raspberry Pi Pico are sent via serial port communication. We also use the serial port for status and debug messages, so we had to

differentiate the different types. Therefore, we decided to add a prefix to each message type: “D: ” for debug and status messages and “R: ” for messages we want the Pico to read. In this way the Pico can parse the message on the serial port and if the message string begins with “R: ” it will execute the instruction. The next letter of the message indicates which action to perform, “h” is for the hugging air chamber and “w” is for the warm heating in the jacket. The next number in the message indicates whether the action should activate or deactivate, “1” for activation and “0” for deactivation.
An example of an instruction message can look something like this: “R: h 1 ”, which in this case would turn on the air chamber.
During testing we noticed that sometimes the messages can’t be correctly parsed from the Pico if a lot of them are sent together. To solve this issue we added some delay before sending some critical messages.

Obtaining the credentials (Bluetooth vs RFID)

Initially we wanted to retrieve the credentials from the RFID reader as we do with Giffy but we encounter some problems in the import of libraries and in finding free pins to connect to it from the boards so we decided to switch to Bluetooth connection. The bluetooth module was connected to the Raspberry Pi Pico. As for the other functions, the Pi Pico only acts as a slave and based on what it is requested by the master, it tells the user what actions to take.

Saving the credentials

To avoid the user entering credentials every single time, we decided to save the last valid credentials.
The credentials are saved in a text file into the Pico and when they are requested the microcontroller reads them from the file and sends them to the other microcontroller. If the credentials are valid it will connect to the network otherwise the user will be asked again. If the new credentials are correct they will be saved in the file instead of the old ones and used at the next use.

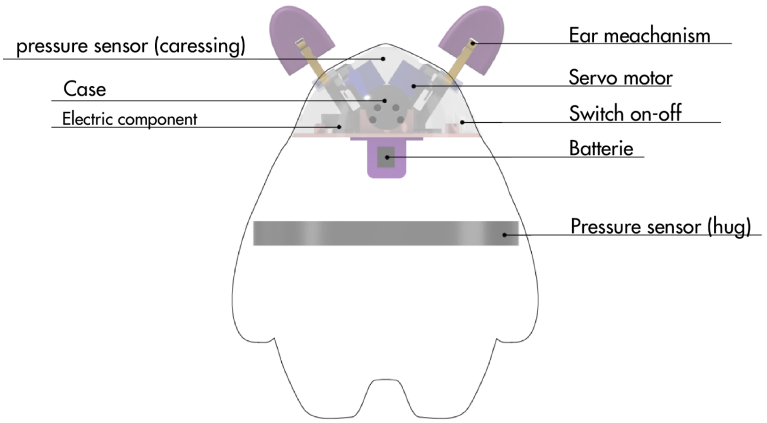
Initially the idea was to use the EEPROM memory on the Huzzah, but we discover it is not present so we tried to save data in flash memory before discovering after trying different libraries that saving a file is much more easy on the Raspberry Pi Pico.

2.6 - Structure

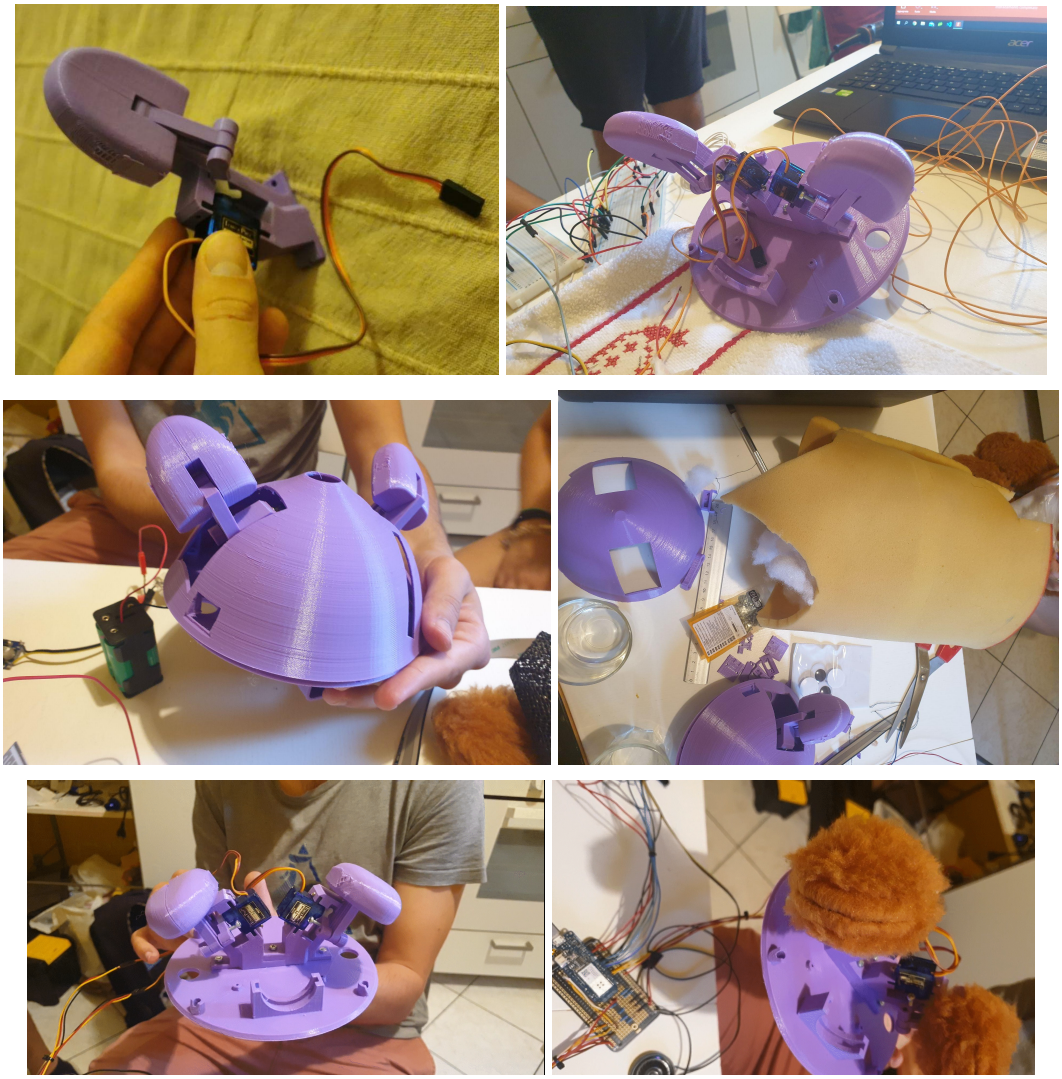
2.6.1 - Giffy

Giffy is mainly made of the following components:

- The body: is furry huggable stuffed with polyester fiber holding all the components together.
- Ears: When it is stroked, it bends and straightens in response.
- A switch: situated so that the user may manage it.
- A speaker: makes varying sounds in response to caresses.
- Pressure sensor 1: located on the head and wirelessly connected to the vibration motor on the jacket.
- Pressure sensor 2: located around the belly and wirelessly connected to the air chamber on the jacket.



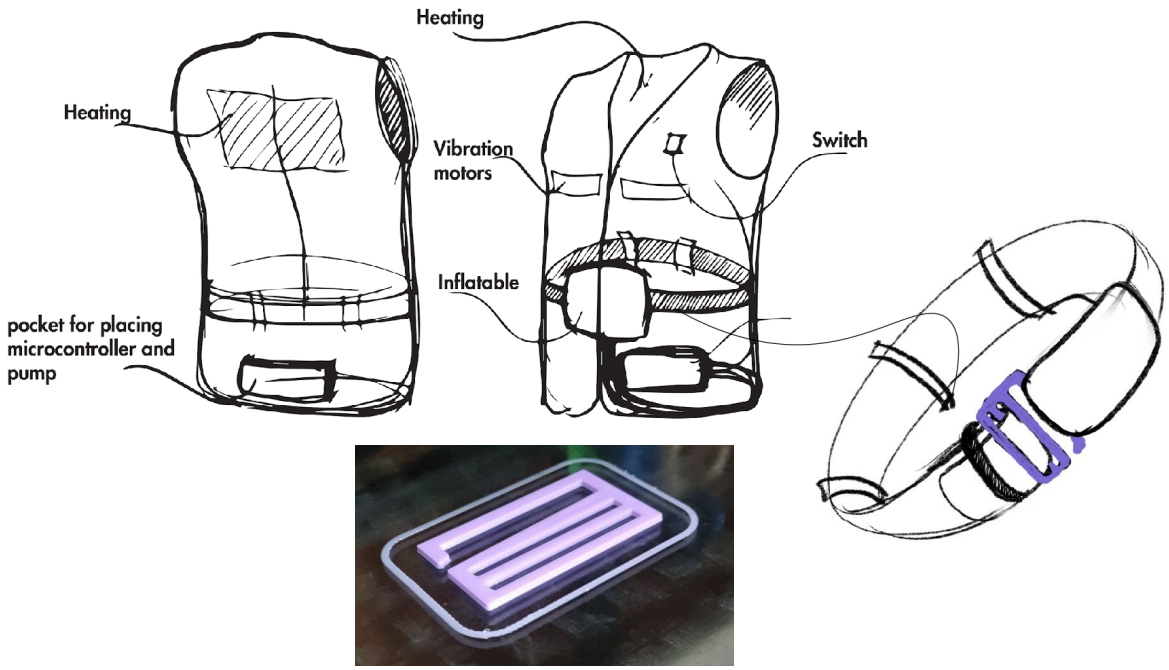
In Giffy the main portion is the head, In the upper part of the head, there is a PLA shell that gives shape to the head and ears. Inside are the batteries, electronic components, and the mechanism that sets the ears in motion. The body is furry and huggable, and it's stuffed with polyester fiber to keep everything together. The ears bend and straighten in response to being stroked. A switch is a device that allows the user to control it. In response to caresses, a speaker makes a variety of sounds. Pressure sensor 1 is mounted on the head and is wirelessly connected to the jacket's vibration motor. Pressure sensor 2 is located around the belly button and is wirelessly connected to the jacket's air chamber.



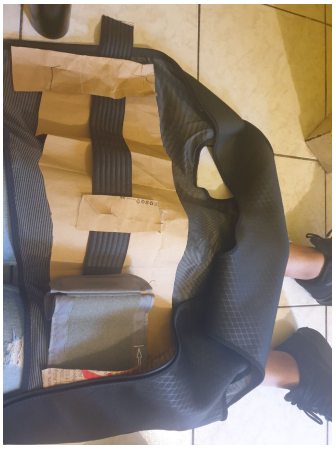
2.6.2 - Jacket

As shown in the picture below the Jacket is the second main component of the P'Hug, It mainly contains:

- Heating vest: to convey a constant warm sensation.
- Inflatable chamber: to convey a hug.
- A switch: users can switch the jacket on and off from here.
- A storage box: to put an air pump, microcontrollers, and voltage regulators.
- Vibration motors: located on the shoulder and chest area responds to caressing from Giffy.
- Elastic strap: for tightening or loosening the hug.

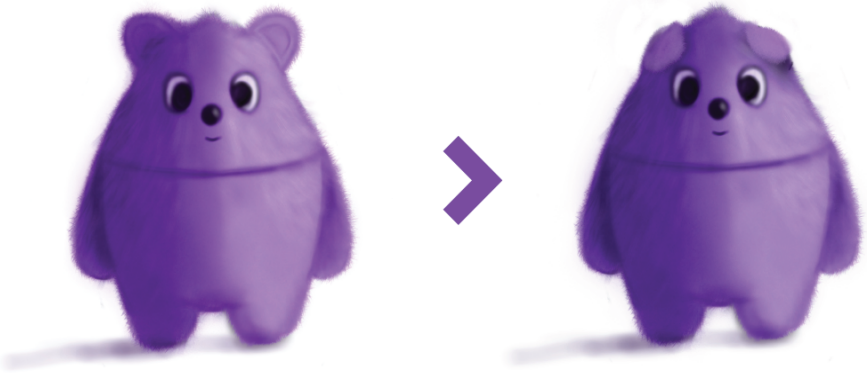


Our main challenge in this part is to be able to have all the functionalities work together and control them. At this stage, we cut the jacket apart and try to figure out the system inside in order to locate the heating wires and where to put the rest of the wires.



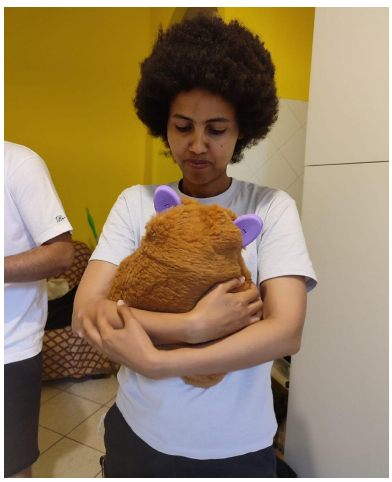
2.7 - Shape

Giffy is a chubby stuffed toy that loves to be hugged. At first, we thought of using a simple stuffed animal like a teddy bear, but later we thought it would be fun to do our design. That way it will be easier to install our devices where we want them and how we want them.



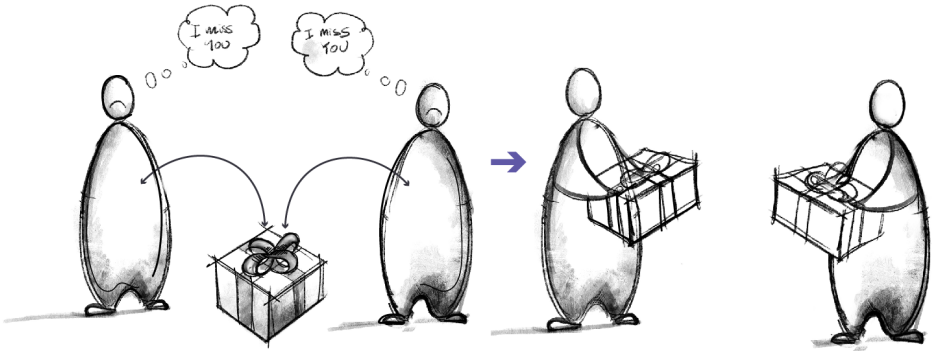
Giffy moving ears

We chose a soft “furry” material for the external part. We put everything together and placed the interior structure of the head inside as well. To give Giffy the soft, huggable and at the same time stable shape we tucked it with polyester fiber.

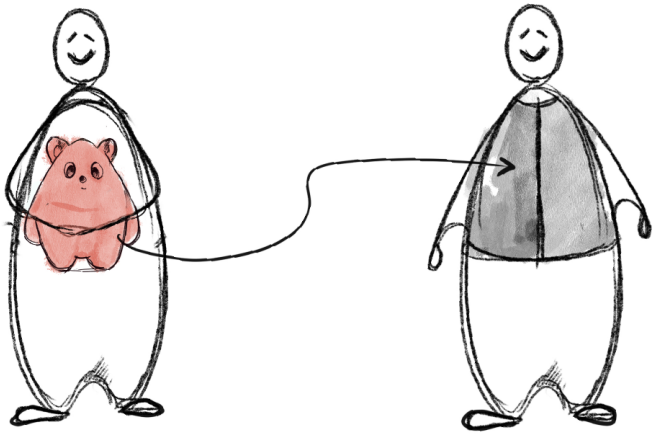


2.8 - Concept

The main idea is to use haptic technology to create emotional interaction. The goal is not to replace a real embrace, but to provide a closer sensation of being there for people who need affection. Its purpose is to provide some sort of mutual connection.



Two people give each other gifts, and the P'Hug comes in two parts. One of them receives the Jacket the other receives Giffy.
Person 1 caresses Giffy ➔ Person 2 receives a vibration sensation.
Person 1 hugs Giffy ➔ Person 2 receives a hugging sensation.
Person 1 shakes Giffy ➔ Person 2 receives a vibration sensation.



Phase 3: Deliver - Final Robot Description

In this part we will describe the final product, its shape, mechanics, electronics and code.

3.1 - Strategy

The goal for the final delivery is to figure out how to combine all the things tried so far. Trying components individually or even in small groups can lead to errors that are not immediately visible in the final assembly.. This applies to both software and hardware. In addition, soliciting final 3D prints over time also led to understanding what the strengths and weaknesses of the design were.

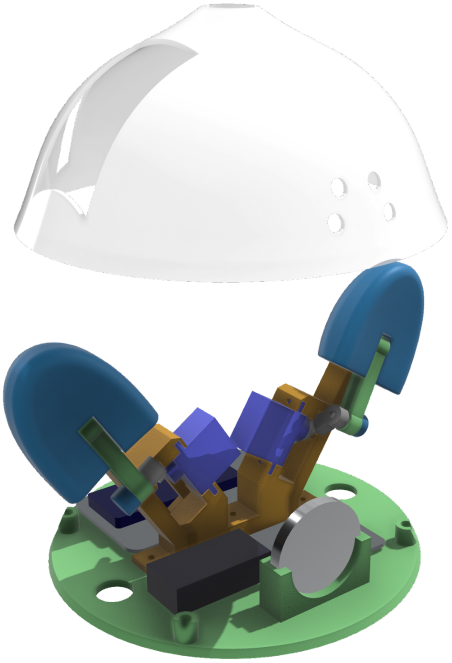
3.2 - Shape

The shape of the puppet is abstract and does not recall any real animal so that the user can use their imagination to see the partner to hug. It comes with curved and rounded shapes that invite the user to experience affectionate feelings with a consequent desire to hug him. It is composed of a soft and furry brownish fabric that contains padding in synthetic material and a PLA shell where mechanisms and electronic components reside. The soft fabric is assembled in two parts shaped with scissors and then sewn by hand. The synthetic material inside is soft material together with polyurethane foam on which the sensor that perceives the user's hugs is wrapped.

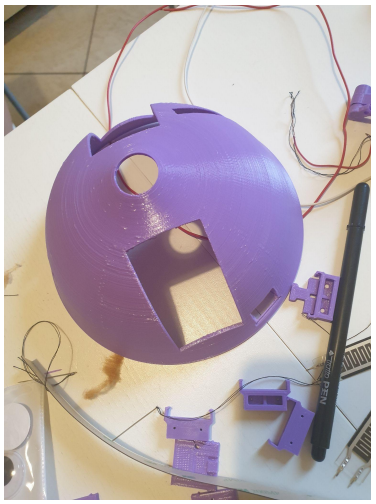
In the upper part of the plush, there is a PLA shell that gives shape to the head and ears. Inside are the batteries, electronic components, and the mechanism that sets the ears in motion.

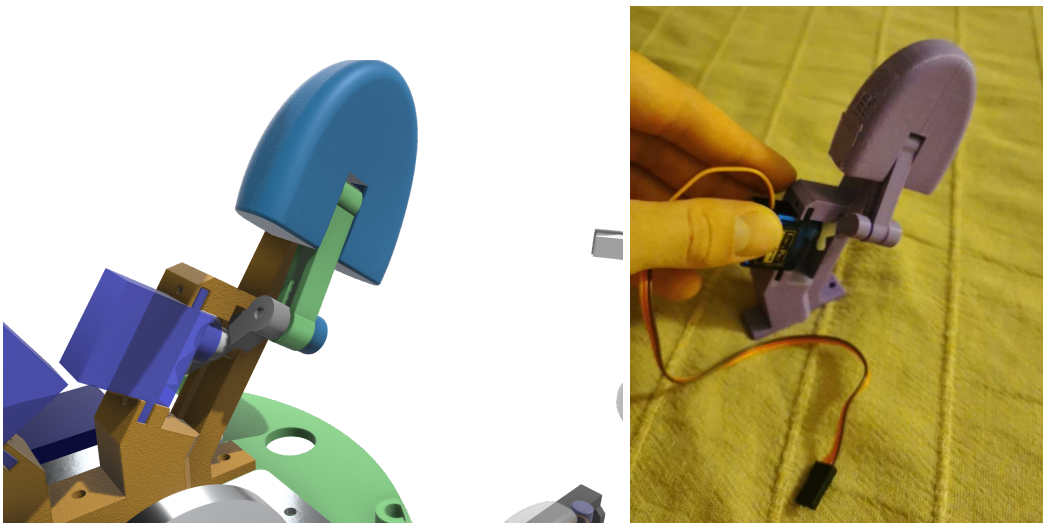
3.3 - Mechanics

The head is made up of two bodies: the dome and the plate. The dome is used to shape the puppet, to be able to caress it on the back, and to protect the electronic materials. The plate, on the other hand, is the base on which the electronic components such as the speaker, the on-off switch, the pcb and the ear mechanism are fixed.

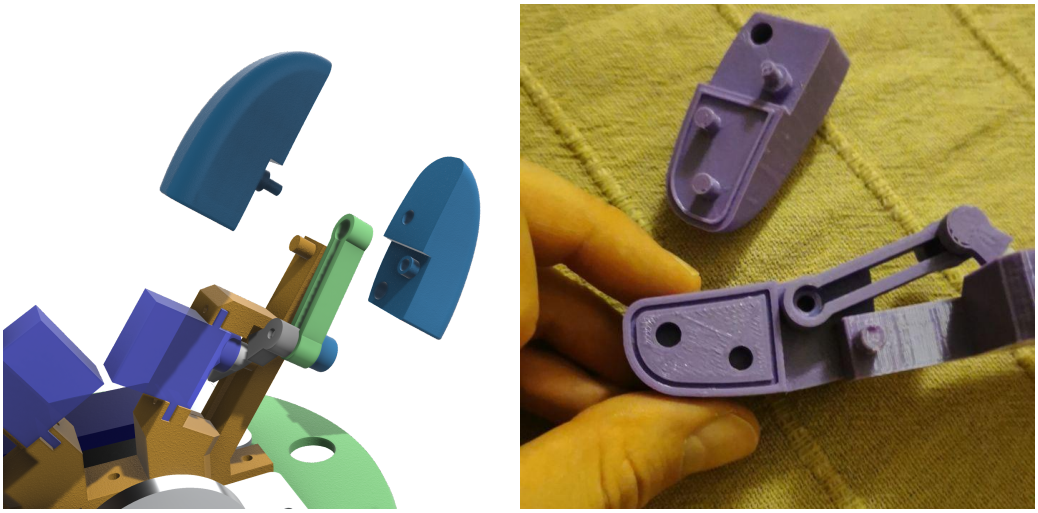


The ears can move by making a rotation around an axis. The rotation is given by a connecting rod that connects the ear and the servomotor.

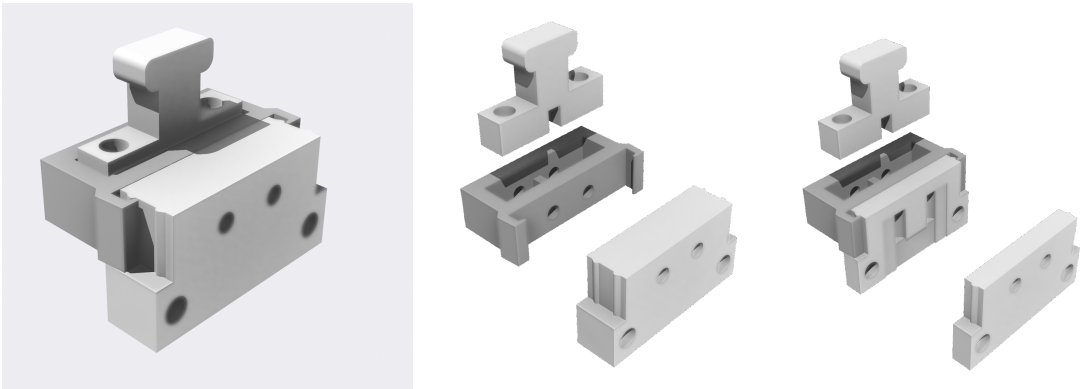




The ear is made up of two parts, which fit together and hold the whole mechanism together.



In the rear part of the dome, we find the pressure sensors that receive the caress signal glued. These, being difficult to solder, were joined to the wires powered by the batteries through a joint designed ad-hoc that allows the cables to reach the sensor at a 90-degree angle. The joint is made up of two parts that attach via snap-fit.

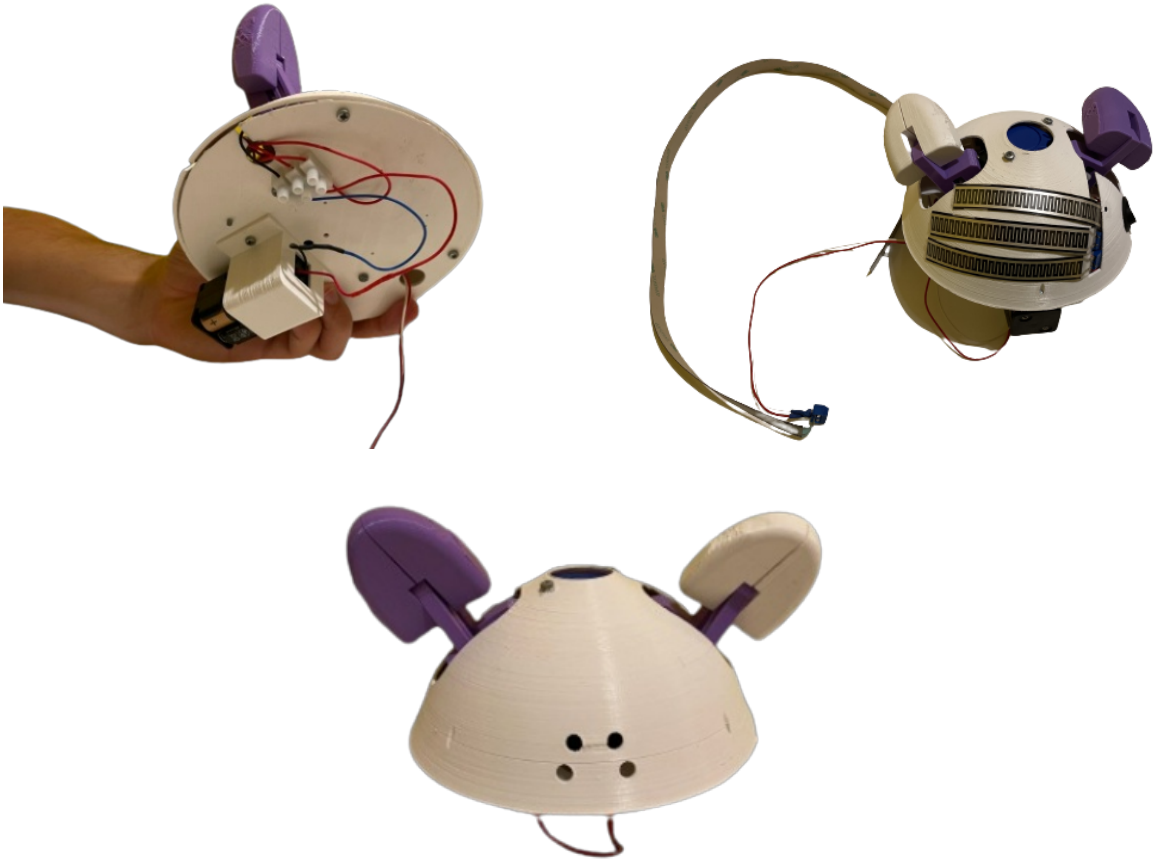


Printed connectors

We designed connectors for the four force sensor resistors. Even if the printed ones are more customised and more suitable, they are not as reliable as the KF301 connectors. For this reason we decided to not use our printed force sensor resistors.

KF301 connectors

Here you can see the final structure of the puppet's head.

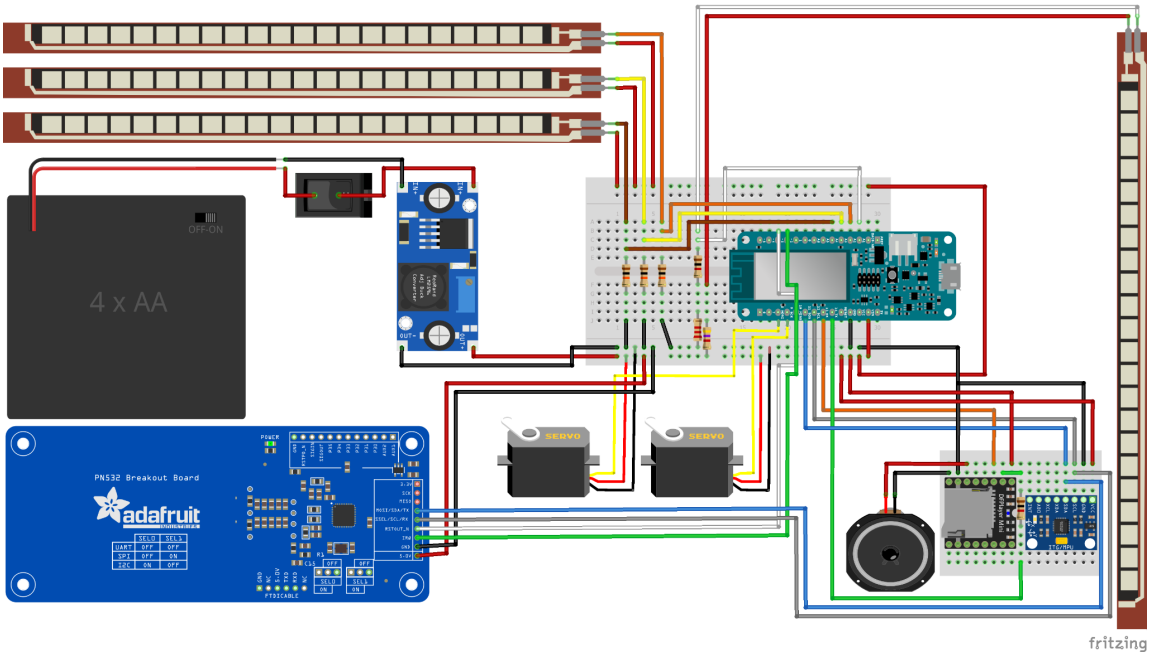


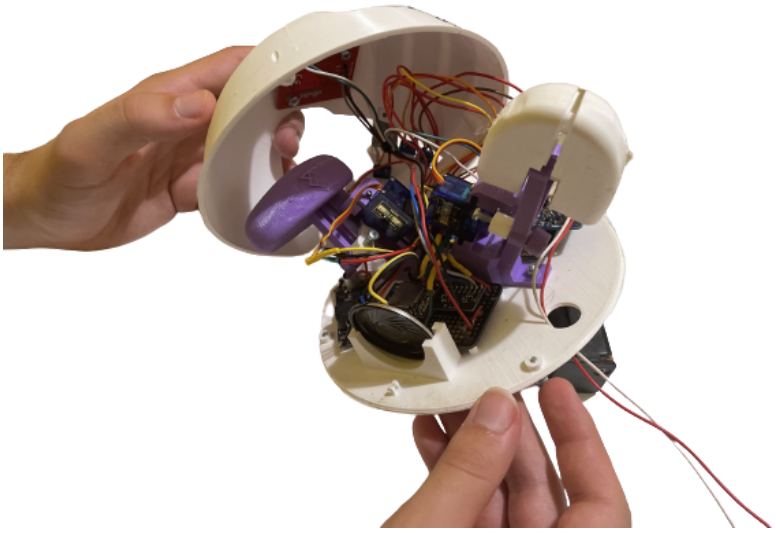
3.4 - Electronics

Once the testing phase was finished, we made wiring diagrams to have an accurate diagram of the connections. To carry out the permanent connections, we decided to use electro cookies, which are permanent breadboards.
It was not easy to carry out the connections because they had to be tinned and the space was not much. In addition, we also tried to optimise the layout of the wires to have a neat board that would allow us to understand any errors.

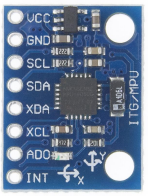
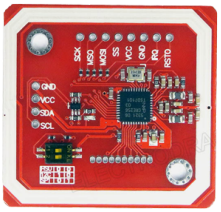
3.4.1 - Giffy

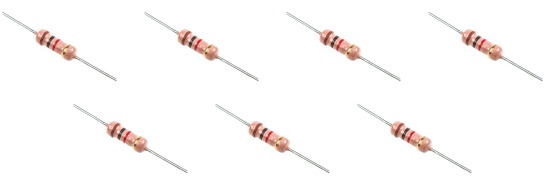
Wiring diagram





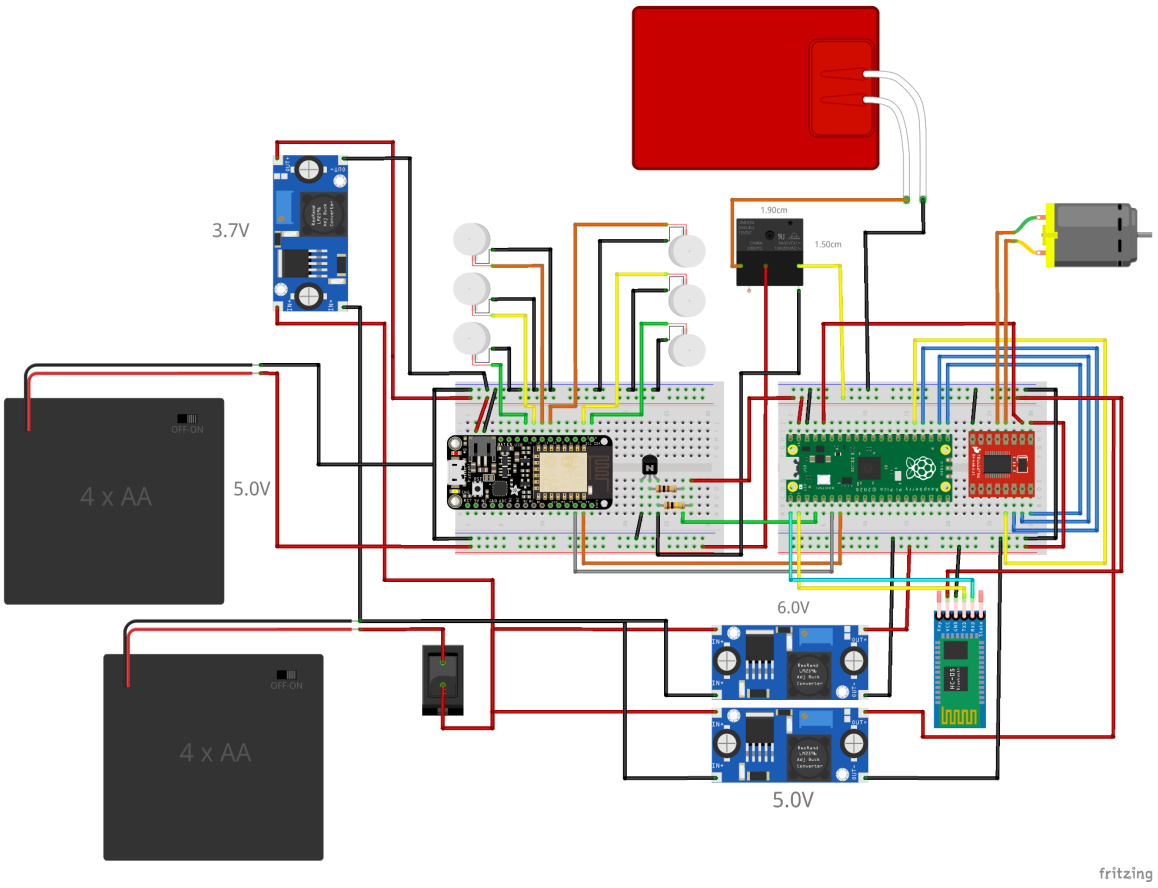
Components

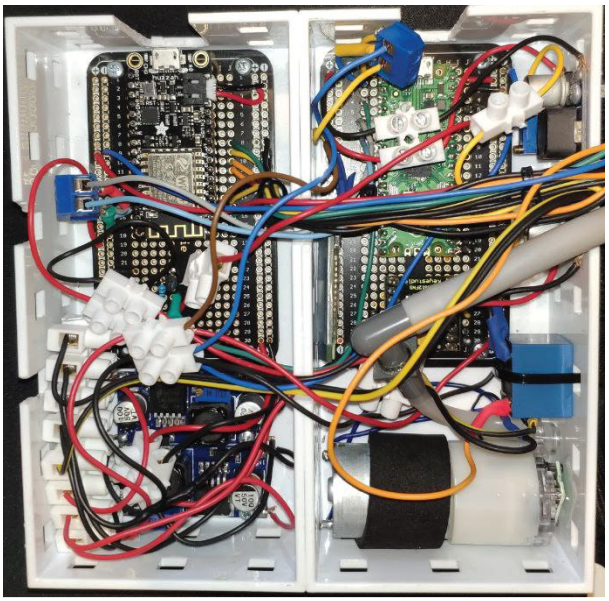
Arduino MKR1000	
3 x RP-L-110 force sensing resistors which are used to perceive the caresses	
Extra-long force-sensitive resistor Interlink 408 in order to perceive the hug.	
GY-521 MPU6050	
ICQUANZX PN532	

NFC TAG	
KeeYees Mini MP3 DFPlayer Player with 2W speaker	 
2 x Micro Servo SG90 in order to implement the ears of the puppet	
Yizhet Converter Buck DC-DC Step Down Module	
Battery holder 4xAA	
Switch	
7 x Resistors (1kΩ - 2kΩ - 5kΩ - 10kΩ)	

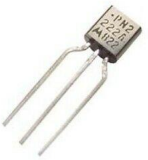
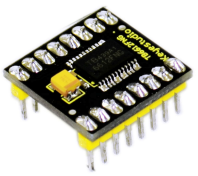
3.4.2 - Jacket

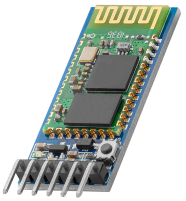
Wiring diagram





Components

Adafruit Feather HUZZAH ESP8266	
Raspberry Pi Pico	
Mini vibration motors	
2 x Resistors (100Ω - 100kΩ)	
Transistor PN2222	
Ks0066 keyestudio TB6612FNG Motor Driver	
Air pump	
3 x Yizhet Converter Buck DC-DC Step Down Module	

Battery holder 4xAA	
Power Bank USB	
Inflatable chamber	
Relay 5V	
Heating vest	
Switch	
HC-05 (ZS-040) Bluetooth Module	

3.4.3 - Bill of Materials

GIFFY				
Object	price	quantity	Total price	Link
Arduino MKR1000 Wifi (Gift)	€ 0,00	1	€ 0,00	https://tinyurl.com/3xkbnuu8
Cloth	€ 10,00	1	€ 10,00	
Battery holder	€ 7,30	1	€ 7,30	https://tinyurl.com/3uw6usdp
Voltage regulators	€ 9,98	1	€ 9,98	https://tinyurl.com/mtym2cc5
Big pressure sensors	€ 27,00	1	€ 27,00	https://tinyurl.com/nhv5njzn
Pressure sensors for caresses (broken)	€ 8,08	1	€ 8,08	https://tinyurl.com/4z7szw7r
Pressure sensors for caresses (new)	€ 11,00	3	€ 33,00	https://tinyurl.com/3t3d464f
Servo motors	€ 5,00	2	€ 10,00	https://tinyurl.com/muv7ayyu
Accelerometer	€ 7,00	1	€ 7,00	https://tinyurl.com/3sc3fyhh
Speaker	€ 10,99	1	€ 10,99	https://tinyurl.com/2p2nwp26
wires	€ 20,00		€ 20,00	

JACKET				
Adafruit Feather HUZZAH with ESP8266	€ 21,00	1	€ 21,00	https://tinyurl.com/s7e6urbt
Raspberry Pi Pico	€ 4,50	1	€ 4,50	https://tinyurl.com/nz3mmfyk
Heating Jacket	€ 30,00	1	€ 30,00	https://tinyurl.com/4z5jyk7u
Battery	€ 20,00	1	€ 20,00	
Sphygmomanometer	€ 40,00	1	€ 40,00	https://tinyurl.com/mtswn5wv
Vibration motors	€ 1,24	10	€ 12,40	https://tinyurl.com/5yvk5tyc
Consumable (small components to test and cable management...)	€ 30,00		€ 30,00	
PLA	€ 0,00		€ 0,00	

TOTAL			€ 301,25	

3.5 - Coding

We created a GitHub repository in which we shared code and we used PlatformIO, an environment which speeds up and simplifies the creation and delivery of embedded products. We used this tool in order to write code which is more reusable and at the same time cleaner and easier to understand.

We decided to use an object oriented paradigm and a client server architecture in which the puppet is the client while the jacket is the server.

The first target we set was to implement the communication between the two entities. The client elaborates the information coming from its sensors and sends a message according to the action performed. The message exchanged between client and server is a simple character which indicates the action to be performed in the jacket.

The application has two main scripts:

- **client** (which run on the puppet)
- **server**(which run on the jacket)

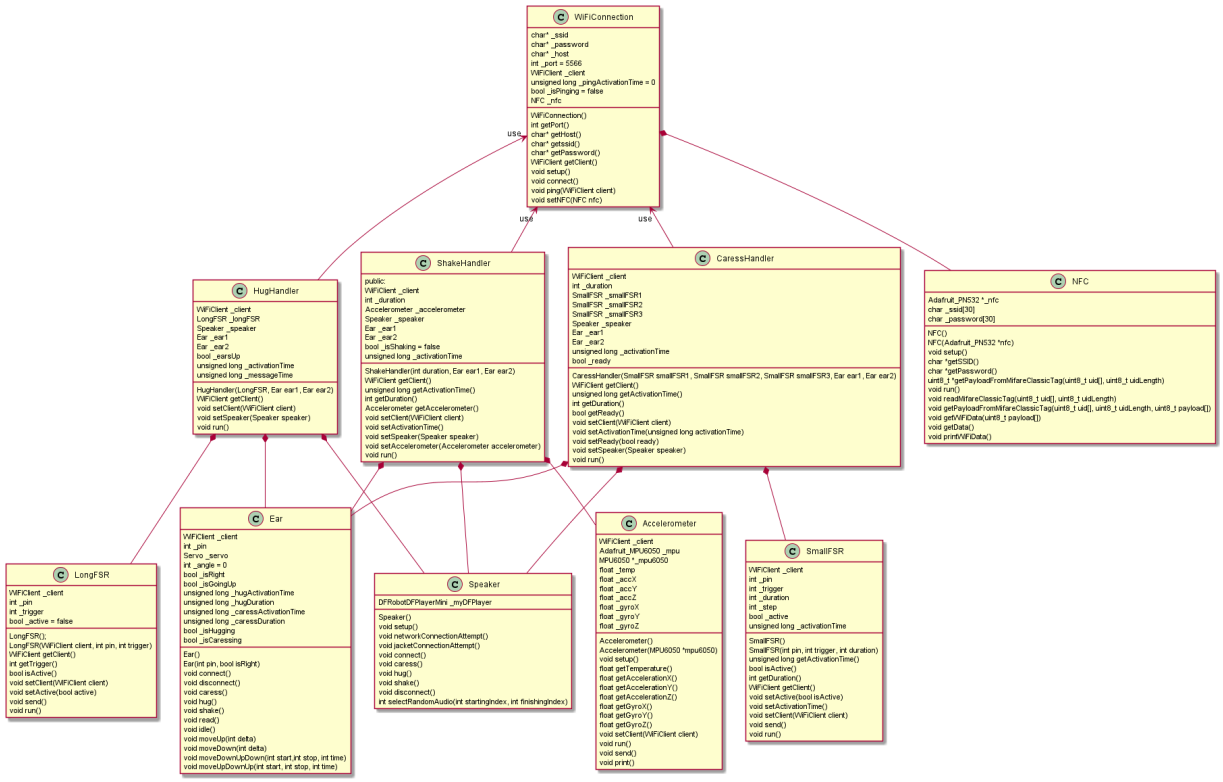
We focus now on the client and then we will present the server

3.5.1 - Giffy

The client.cpp has two main functions:

- setup() in which the objects are created and initialised.
- loop() in which we initialise a connection. Until the connection is stable, the flow of execution enters in a loop function called run() which is in charge of calling the three main run functions of the handlers. Once the connection is lost, the flow of executions starts another time from the starting point of the loop.

We are using a bottom-up approach in order to describe the components of the client entity. Below you can see the UML diagrams of the classes of the client.



https://github.com/AlessandroBarbiero/Robotics-and-Design-project/blob/main/Client/out/diagrams/diagram/diagram.png

As you can see from the UML diagram we implement a specific class for each type of sensor/actuator module.

Class	Sensor/Actuator
Accelerometer	GY521
Ear	Micro Servo SG90
LongFSR	Extra-long force-sensitive resistor (FSR) Interlink 408
NFC	NFC PN532
SmallFSR	RP-L-110
Speaker	mp3-tf-16p

Sensor classes are in charge of reading values from the analog pin while actuator classes are in charge to perform the actions.

We then implemented three main handler classes in order to perform the three main functionalities of the puppet:

- HugHandler.cpp
- CaressHandler.cpp

- ShakeHandler.cpp

Each of those handler classes have a similar structure. They have sensors and actuators classes as attributes which are used to perform the respective actions and a main run() function which is directly called from the main which performs the action (read sensor, move actuators and send messages to the jacket).

```
void loop() {

  speaker.jacketConnectionAttempt();
  delay(5000);

  Serial.print("connecting to ");
  Serial.println(wifi.getHost());

  if (!client.connect(wifi.getHost(), wifi.getPort())) {
    Serial.println("connection failed");
    return;
  }

  Serial.println("Connected");

  speaker.connect();
  ear1.connect();
  ear2.connect();

  caressHandler.setClient(client);
  hugHandler.setClient(client);
  shakeHandler.setClient(client);

  while(client.connected()){
    wifi.ping(client);
    caressHandler.run();
    hugHandler.run();
    shakeHandler.run();
    delay(CLOCK_INTERVAL);
  }

  Serial.println();
  Serial.println("closing connection");

  speaker.disconnect();
  ear1.disconnect();
  ear2.disconnect();
}       Lorenzo Poretti, mese scorso • WiFiConnection clas
```

HugHandler is a class which handles hugs, in particular it has a force sensor which is activated when the pressure is above a certain pressure value. The speaker emits kiss sounds and the ears move alternatingly. A ‘h’ character is sent to the jacket via WiFi.

CaressHandler is a class in charge of handling caress. Sensing caresses is a little bit harder with respect to sensing hugs. There are three force resistor sensors on the head which trigger the CaressHandler when the sensors are activated in a specific order (from the higher to the lower) and the activation time between one touch and another one is not expired. Once the caress is sensed the speakers emit relaxing sounds and the ears are lowered. A ‘c’ character is sent to the jacket via WiFi.

ShakeHandler is a class which handles shaking actions. The inertial measurement unit (IMU) registers accelerations and angular velocities. If the velocities remain above a certain value for a given time, a ‘s’ character is sent to the jacket, a shout is emitted from the speaker and the ears move fast.

The handlers in order to communicate with the server use a **WiFiConnection** class. This class, in addition to performing the connection, pings the jacket and it handles the disconnection from the server. The **NFC** module has an important role in the connection as it is in charge of reading the parameters of a new network such as SSID and password. The parameters of the current network are saved in the NFC tag for future readings.

LIBRARIES USED

We used the following external libraries:

- Arduino.h
- Servo.h
- WiFi101.h
- SPI.h
- Wire.h
- Adafruit_MPU6050.h
- Adafruit_Sensor.h
- DFRobotDFPlayerMini.h
- Adafruit_PN532.h

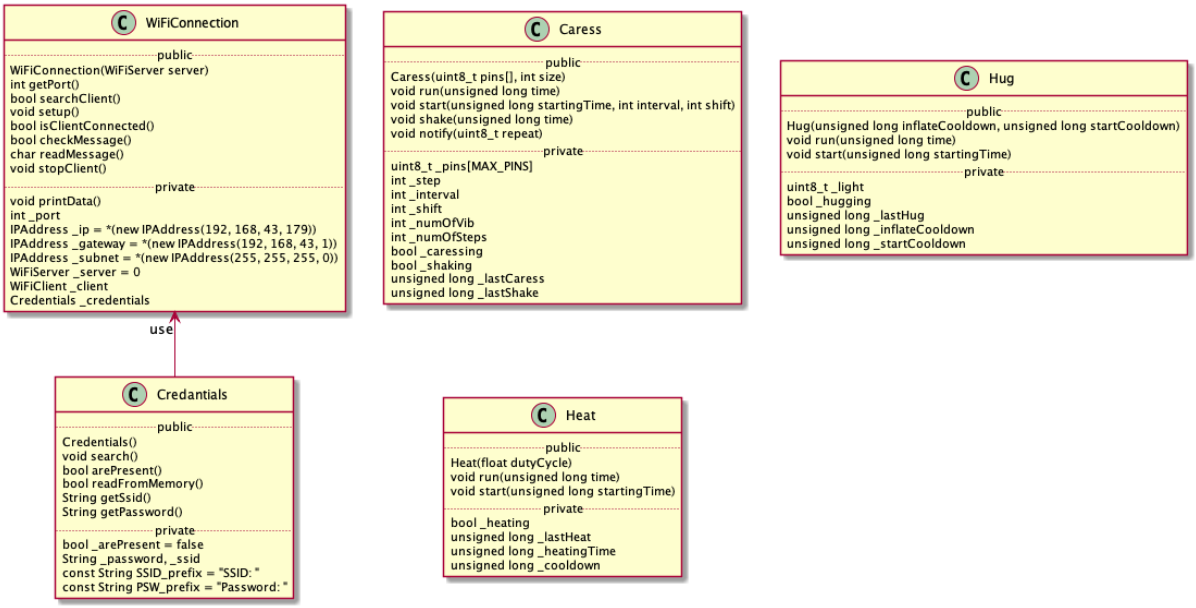
3.5.2 - Jacket

The server.cpp file has two main functions:

- setup() in which the wifi connection is initialised reading the credentials from the memory or from the Bluetooth connection.
- loop() in which we wait for a client to connect. Once a connection is performed, the flow of execution enters in an internal loop until the connection is guaranteed. In the internal loop the run() functions are called, and the messages are read. Each message type triggers a specific start() function. Once the connection is lost (detected with a custom made ping system), the flow of execution starts another time from the starting point of the loop, searching for a new client.

All the messages sent for debug purpose starts with “D: ” to distinguish them from the command messages sent through the same channel to the Raspberry Pi Pico, starting instead with “R: ”

You can see below the UML diagrams of the classes of the server.



https://github.com/AlessandroBarbiero/Robotics-and-Design-project/blob/main/Server/Server%20Huzzah/diagrams/server.png

For the server, we have three different classes for actions performed on the jacket:

- Caress.cpp
- Hug.cpp
- Heat.cpp

Caress is a class that activates the vibration motors in a waveform manner, in order to convey the sensation of a caress. The wave of the vibrations is created by defining a duration of the vibrations and a shift describing the delay between vibrations. This class also generates the action of a shake.

Hug is a class that inflates the air chamber by activating the air pump for a set amount of time.

Heat is a class that activates the heating system in the jacket.

All server classes have a set cooldown that defines how often they can be activated. The server only calls a start() function when the class cooldown is over. This is useful in order to not overwhelm the user with sensations, or for example inflate the air chamber too much and break it.

The server also has a **WiFiConnection** class, which connects to a WiFi network, initiates the server and waits for a client to connect. While a client is connected, the server continuously checks for a ping message from the client. If a ping is not received after a set amount of time, the server reverts back to the client searching state.

The **Credential** class is used by the WiFiConncetion class to handle the read of the credentials from the memory or from the Bluetooth connection with the user. In order to do so it comunicates always with the Raspberry Pi Pico asking him to read saved data or to search new ones.

LIBRARIES USED

We used the following external libraries:

- ESP8266WiFi.h

We used a Raspberry Pi Pico because one controller was not enough. We used MicroPython to program it. This controller acts as a slave and reads from the serial port the messages sent by the master. The messages are parsed and depending on the command sent it performs a different function.

The Raspberry Pi Pico has a bluetooth module from which we can read the Wi-Fi network credentials to allow the jacket to connect.

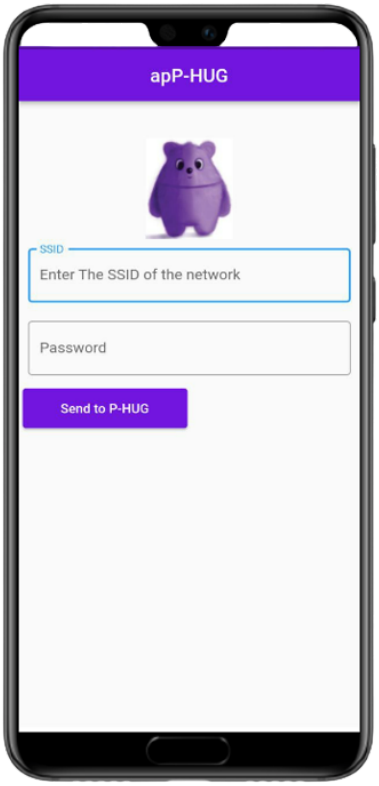
The code is organised in two loops. The first one is used to read the Wi-Fi credentials sent by the user via app. The second loop reads the messages coming from the master.

When received, the message is decoded into utf-8 and then split.
If the first word is R: the message is intended for the pico and is read.

3.5.2 - Apps

We implement a mobile application for the puppet to send network data over NFC.
We decided to use Flutter, an open-source UI software development kit created by Google. The interesting thing about Flutter is that it is possible to code both Android and iOS applications.

The application has two text fields where you can enter the data related to the network SSID and its password.



The data is sent using NFC to Giffy putting the mobile near his head and pressing the send button.
You can download the Android app from this link:
https://github.com/AlessandroBarbiero/Robotics-and-Design-project/blob/main/App/APK/apP-HUG.apk

You can also use an external NFC app selecting Plain Text and putting SSID*PASSWORD as PlainText. We suggest to use this app:
https://play.google.com/store/apps/details?id=com.gonext.nfcreader&hl=it&gl=US

In order to configure the network for the jacket we decided to use an external application which sends data over bluetooth.
You can choose this one if you want to connect from an Android device:
https://play.google.com/store/apps/details?id=de.kai_morich.serial_bluetooth_terminal&hl=it&gl=US

Or this one if you are trying to connect from a Windows device:
https://apps.microsoft.com/store/detail/bluetooth-serial-terminal/9WZDNCRDFST8?hl=it-it&gl=US

In every case make sure to first pair the device HC-05 from the device you want to talk from. If a password is asked, put “1234”.

Conclusion

We would like to mention some of the major lessons this course has given us:

- Hardware is not software, it can be easy breakable and if you break it you have to pay
- Engineers approached the design field as designers approached the engineering field learning a lot from each other
- Communication is extremely important in a team. It’s fundamental to divide tasks and organise activity
- Flowcharts and other types of diagrams seen in class are more important than we expected. Diagrams help us a lot during design and implementation.
- Know new people and discover different cultures
- During both hardware and software development it’s better to solve problems that arise as soon as possible and don't let them piling up
- Try to maintain things as easily as possible during the design phase and not try to implement the best customizable thing ever.
- Keep the curve complexity- emotion as low as possible improving emotion without improving complexity
- Modularity is the way, order and reusability in the code is extremely important
- Programming a microcontroller, read values from sensors and handle actuators
- Understand how to express emotion in a robot and understand how to design and program it in order to interact with people
- We aimed a little high when it comes to functionality. This meant we had to put a lot of hours into the project in order to achieve a result we were happy with.

The relationships of the group were excellent and everyone always tried to be as useful and helpful as possible, contributing with one's experience. It was not difficult to divide the work, but mostly we worked together in groups of at least 2.

During the project we had to make several shopping trips and online orders for part. This was due to us not really knowing what to buy at all times. Sometimes we tried a part and it didn't work as we intended and sometimes parts were broken. In the end, our expenses were over our original budget.

APPENDIX

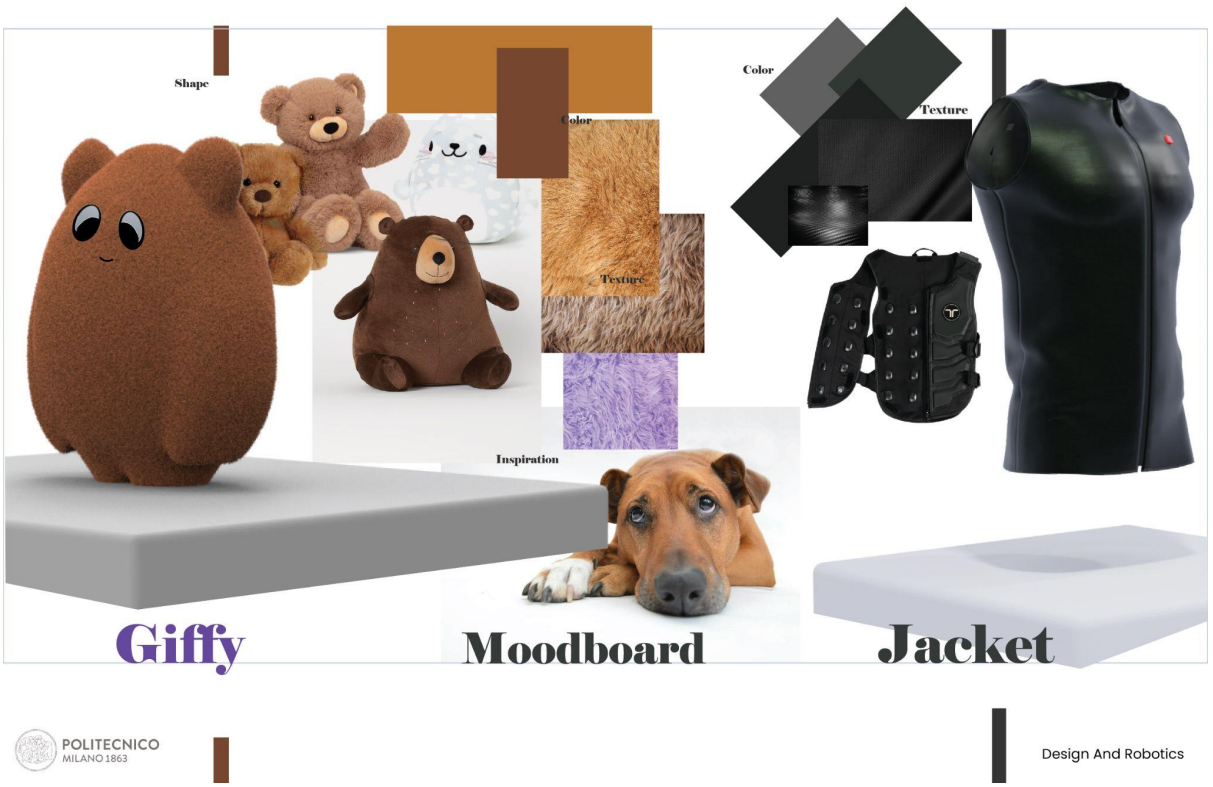
A - User Manual

User Manual

B - Maintenance Instructions

P'Hug Maintenance

C - Moodboard

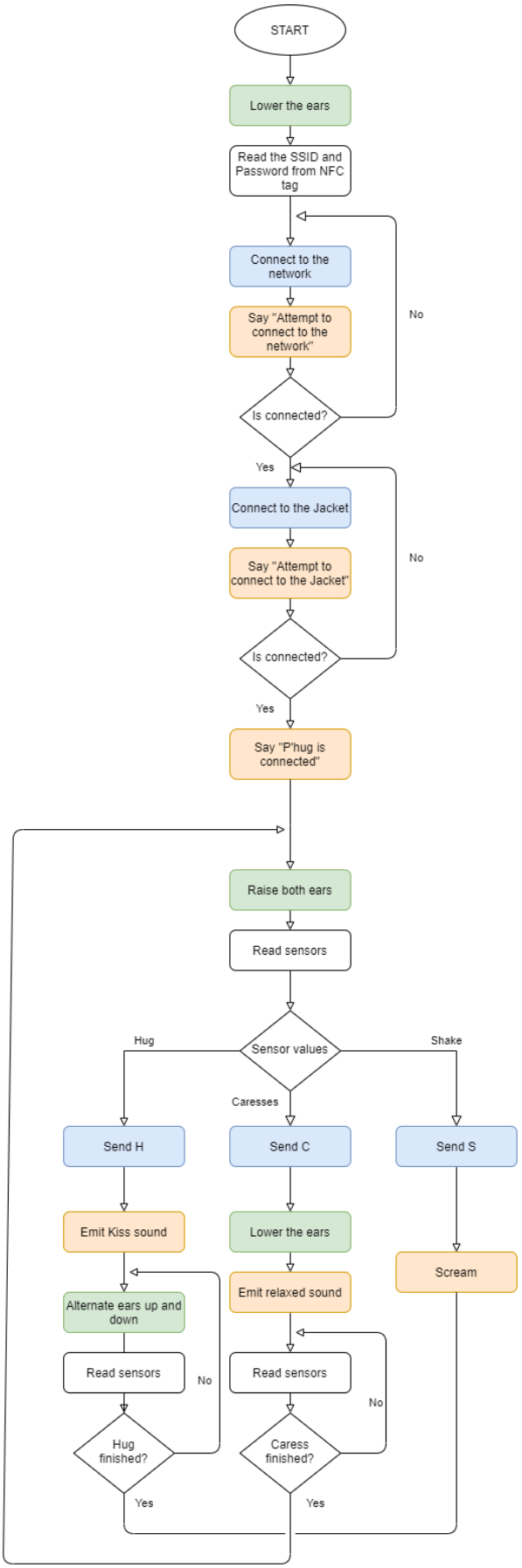


D - Datasheets

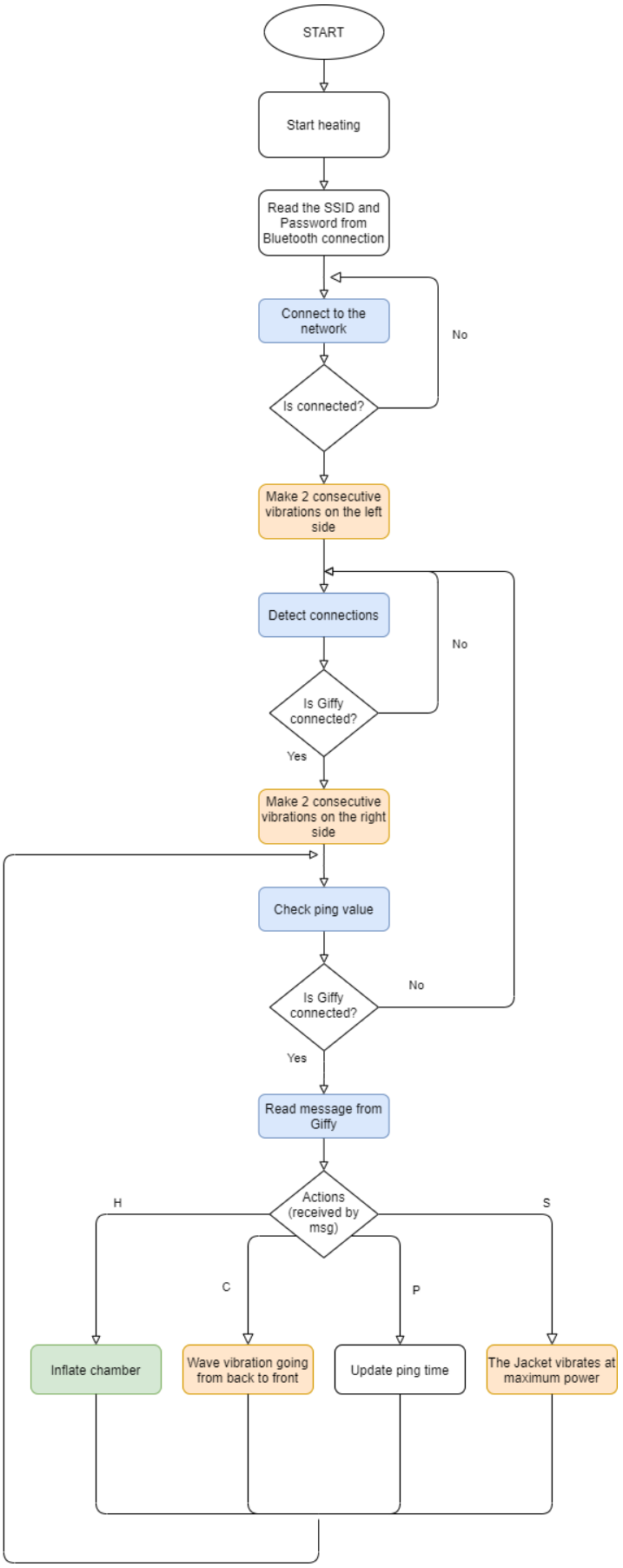
Datasheets folder: Datasheets

E - Interaction Flowchart

Interaction flowchart for Giffy:



Interaction flowchart for the jacket:



Minutes of the Meetings

Meeting 29/03/2022

Time: 9.15 - 17.15
Venue: Lab prototipi
Attendees: (everyone)

- First lecture
- We discussed our first ideas and tried to find the one to develop
- To help us brainstorming our ideas we exploited the Miro.com website and application
- At the end of the day we decided about a trash eater that can perceive near objects and detect emotions with image recognition

Meeting 5/04/2022

Time: 9.15 - 18:00
Venue: Campus Bovisa
Attendees: (everyone)

- We discussed emotions that the robot can detect and express.
- We constantly search for new ideas always using miro.com
- We bring on and develop together different ideas, everyone with his pros and cons
- Ended the meeting without a major project but a lot of ideas

Meeting 12/04/2022

Time: 9.15 - 19:00
Venue: Campus Bovisa
Attendees: (everyone)

- We tried to define the robot we want to make, functionality and aesthetics.
- We found four main ideas and decide among these through a votation on a blackboard defining pros and cons of each solution
- I-Hug won
- Defined the common functionality that our prototypes should convey, writing down all inputs and outputs
- We decided to leave freedom for the interpretation of the prototypes

Meeting 14/04/2022

Time: 14.00 - 19:00
Venue: Discord
Attendees: (everyone)

- We prepared the presentation defining the general idea.

- The Engineers write down a first draft that then Helen fixed and made aesthetically pleasant

Meeting 15/04/2022

Time: 17.00 - 19:00
Venue: Discord
Attendees: (everyone)

- We defined and wrote down all the possible interactions of the robot to help us understand all the possible scenarios the robot can be involved into

Meeting 26/04/2022

Time: 9.15 - 18.00
Venue: Lab prototipi
Attendees: (everyone)

- Second lecture
- Bring prototypes
- Starting from a hugging robot with arms and image recognition (I-Hug)
- Thanks to professors advice we refined our idea about the project and switch to the final one (P'Hug)

Meeting 30/04/2022

Time: 9.15 - 21.00
Venue: Luigi's home
Attendees: (everyone)

- After buying some of the components we start our work on the electronic parts
- First works are made on the pressure sensors and wifi connection
- First problems occurs for the choice of the wifi library, we discover that we can't use the same for the two microcontroller because the feather huzzah doesn't support wifi101
- Problems start when we tried to connect two devices at the same network, it seems not working
- We discover how to give static ip to one device
- Start to implement a function to perform a wave like vibration utilising 3 vibration motors
- In this occasion we utilised only a single arduino ".ino" script for each part, the code become to grow

Meeting 03/05/2022

Time: 9.15 - 18:30
Venue: Lab prototipi
Attendees: (everyone)

- Third lecture

- We defined the actions the robot has to perform with the help of a flow chart
- We change the coding strategy, switching to object oriented programming to maintain the code and make it more readable, to do so we change from arduino ide to PlatformIO. The first classes are designed
- The wave function for the vibration motors is improved and now it works, it is parametric in order to find the correct intervals for the vibrations
- The sensor for the caress seems to work, we can now send a char ‘c’ when the caress is performed

Meeting 07/05/2022

Time: 9.15 - 22:00
Venue: Luigi’s home
Attendees: (everyone)

- The long fsr works and we can send a char ‘h’ when the level of pressure goes over a threshold
- We implemented the hug messages, they are correctly sent and received
- Finished the wave sensation for the vibration motors written inside his class
- In order to feel the sensation on the body we build a belt to keep the 3 motor in contact with the body
- When now a caress is performed on one side, the motors can transmit the wave sensation

Meeting 10/05/2022

Time: 9.15 - 21.00
Venue: Lab prototipi
Attendees: (everyone)

- Fourth lecture
- At the end of the lesson smoke comes out of the pressure sensor we used for detecting the caress: it’s broken
- The engineering team stays in the building until closure to figure it out what happened: we used a too low resistor and keep running the sensor for 6 hours
- We have to change strategy, that was the last sensor of that kind ready for shipping
- Luckily we knew what to do from previous research and suddenly ordered the new sensors

Meeting 14/05/2022

Time: 9.15 - 21.00
Venue: Luigi’s home
Attendees: (everyone)

- We tested the new force sensing resistor sensors and we designed a way to use those sensors to sense caresses.
- We wrote code to read sensor values coming from the new force sensing resistor sensors

- We modified the code to send caresses and hugs via WiFi
- We disassembled the sphygmomanometer and we figured out what components were useful for our project
- We measured the voltage of the internal components and we figured out how the sphygmomanometer works
- We decided to remove the pump and to test it

Meeting 17/05/2022

Time: 9.15 - 21.00
Venue: Lab prototipi
Attendees: (everyone)

- Fifth lecture
- We removed the start button form the heating jacket and measure tension and current flowing through it (5V - 1,8A)
- The different levels of heat are implemented changing the frequency
- Talk about disposition of components inside the head of the puppet and the jacket

Meeting 20/05/2022

Time: 22.00 - 24.00
Venue: Discord
Attendees: Alessandro Barbiero, Luigi Altamura, Lorenzo Poretti

- Talk about the report
- Talk about the code
- Talk about the components to buy

Meeting 22/05/2022

Time: 10:00 - 19:00
Venue: Luigi's house
Attendees: (everyone)

- Test components
- Test connection between py pico and feather huzzah
- Work on the report

Meeting 24/05/2022

Time: 10:00 - 19:00
Venue: Bovisa Campus
Attendees: (everyone)

- Test motor driver
- Test connection between py pico and feather huzzah
- Refine 3D printed things

Meeting 26/05/2022

Time: 16:00 - 18:00
Venue: Leonardo Campus
Attendees: Alessandro Barbiero
Luigi Altamura

- Test mosfet for starting the heat of the jacket
- The mosfet IRF730 doesn't work for our purpose: even if we can open the gate we cannot power the jacket through it

Meeting 27/05/2022

Time: 10:30 - 19:00
Venue: Bovisa Campus
Attendees: Lorenzo Poretti
Alessandro Barbiero
Lorenzo Dondi

- Tested ear mechanism, discovered problem in the point of junction between servo and the joint
- Written final wiring diagram for the puppet on fritzing
- Tried to implement the code for the ear movements (set limit angles for start and stop)
- Tested the connector for the fsr
- Tested fsr for the hug on a soft surface simulating a real hug
- Seen where to put electronics inside the 3D printed head of the puppet

Meeting 28/05/2022

Time: 10:30 -20:30
Venue: Luigi's home
Attendees: (everyone)

- Sewed the body of the puppet, a zip to close it and filled it with soft material
- Made the final soldering for the puppet electronics on the electro cookie following the diagram on fritzing
- Tested the puppet with the server, discovered problems in the code, the client continue disconnecting because a loop is present in the functions avoiding the client to send ping messages
- Tried the 3D printed head inside the puppet with relative movement of the ears
- Continue writing the report and sketches of the final video presentation

Meeting 30/05/2022

Time: 10:30 -20:30

Venue: Luigi's home
Attendees: Alessandro Barbiero
Luigi Altamura

- After a long time working to find a solution to power the jacket finally we found it, using a relay 5V and a transistor we can now control the heat of the jacket from the 3V pin of the Raspberry Pi Pico
- Write the code to handle the regulation of the heat
- Refined the communication between pi pico and huzzah
- Soldered all the final wires for the vibration motors
- We lost one vibration motor in the process because the wires are really fragile, so we decided to add extra heat shrink tubing
- We tested all the functionalities of the jacket together using a debug script (all the electronics works)

Meeting 31/05/2022

Time: 9.15 - 17.00
Venue: Lab prototipi
Attendees: (everyone)

- First Delivery: we present a first prototype showing all the functionalities but without putting it all together, we showed the head of the puppet with the movement of the ear, the sounds, and the sensors on one side and the jacket that can inflate the chamber on the stomach and heat up on the other
- We test the vibration motors attached to the jacket using small pockets to detect the best position to put them
- We regulate the variables to feel the perfect wave on the body
- We search for a solution to solve the bug related to the accelerometer library
- We redesign part of the 3D components to solve the structural problems we have
- We tried to solder some components and we break a switch
- We write down the final wiring diagram for the jacket on fritzing

Meeting 01/06/2022

Time: 14.00 - 19.00
Venue: Lab prototipi
Attendees: (everyone)

- Soldered all the final wires for the puppet. We chose the correct length of cables to be able to fit all components into the head.
- Designed the final 3D components to solve the structural problems we have with the head of the puppet
- We tested all the functionalities of the puppet without connecting it to the laptop and it doesn't work. The batteries were dead and after replacing them everything worked fine.

Meeting 07/06/2022

Time: 09.30 - 14.00
Venue: Lab prototipi
Attendees: Lorenzo Poretti
Elias Insulander
Lorenzo Dondi
Helen Berhanu Tekle

- Tried the new head of the puppet and made adjustments.
- Worked on model of holder for electronics for the jacket.
- Made some parameter changes for the code of the air pump.

Meeting 09/06/2022

Time: 09.15 - 20.00
Venue: Lab prototipi
Attendees: Lorenzo Poretti
Elias Insulander
Lorenzo Dondi
Alessandro Barbiero

- Soldered the NFC sensor for the puppet.
- Tried to import the libraries for the NFC sensor without results
- Experiment with the locations of the vibration motors.
- Adjustments to 3D-printed parts: connectors for force sensing resistors, battery holder, holder for vibration holder
- Restructured report.

Meeting 11/06/2022

Time: 14.00 - 17.00
Venue: Zoom
Attendees: Alessandro Barbiero
Luigi Altamura

- Revised wiring schema for the Jacket
- Started final soldering for the Jacket

Meeting 13/06/2022

Time: 08.30 - 13.50
Venue: Discord
Attendees: Alessandro Barbiero
Luigi Altamura

Lorenzo Poretti

- Discussed about last steps
- Managed the reimbursement procedure
- Written the report

Meeting 14/06/2022

Time: 09.30 - 21:30
Venue: Luigi's home
Attendees: Alessandro Barbiero
Luigi Altamura
Lorenzo Poretti
Elias Insulander

- Continued soldering the electro cookie of the jacket (tried to make it detachable)
- Added mammut to Giffy's head
- Tried to use the new 3D printed connectors for the sensors, they failed
- Sew the vibration motors to the jacket (one breaks up during the process)

Meeting 17/06/2022

Time: 14:30 - 19:30
Venue: Lab prototipi
Attendees: Alessandro Barbiero
Elias Insulander

- Worked on the box for jacket electronics
- Fixed electronics in box

Meeting 18/06/2022

Time: 9.30 - 20.30
Venue: Luigi's home
Attendees: (everyone)

- Fixed vibration motors to jacket
- Assabled jacket electronics
- Updated code for Raspberry Pi Pico
- Tried to finalise as much as possible

Meeting 19/06/2022

Time: 9.30 - 22.30
Venue: Luigi's home
Attendees: (everyone)

- Shot the video
- Soldered Final electronics
- Fixed bluetooth module for the jacket

Meeting 20/06/2022

Time: 9.30 - 00.00
Venue: Luigi's home
Attendees: (everyone)

- Finished the report and all other documentation
- Prepared final presentation
- Edited the video
- Finished the puppet
- Finished the jacket

Total hours of the meetings: ~ 258
Minimum individual work of each person of the group: ~ 50 hours
Total work / person = ~ 300 hours

Bibliography

Here there are guides we used to solve complicated problems.

NFC MIFARE CLASSIC CARD:
https://shop.sonmicro.com/Downloads/MIFARECLASSIC-UM.pdf