Firmwares

We decided to use the Finite State Machine model to implement the Architecture of the robot. In this type of model, each state corresponds to a particular set of actions that Hato performs during its working flow:

- **STATE 0:** initial state in which Hato waits to be put on the head
- **STATE 1:** state called immediately after it is put on the head of a child
- **STATE 2:** listening state. If the other child screams during this phase the robot tries to restore calm
- **STATE 3:** handles the other child's screaming case
- **STATE 4:** state in which the robot expresses the sadness related to what the child said
- **STATE 4.5**: state to handle the FSR sensor for the second phase to know whether the hat is putted in the head of the first child or it hadn't been taken out from the second child's head.
- **STATE 5:** listening state of the second phase: the one in which each child tries to explain the mate's point of view
- **STATE 6:** the equivalent of state 3 for the second phase
- **STATE 7:** the robot expresses its happiness after the child expressed his mate's point of view
- **STATE 8:** the tentacles' leds turn on and the robot expresses its happiness after both of the children have explained the other's point of view
- **STATE 9:** the robot waits for both of the children to squeeze the tentacles
- **STATE 10:** the robot is excited after the children have squeezed the red tentacles
- **STATE INIT1:** state that, by means of flags, handles the transition between one child explanation and the other one
- **STATE RESET:** state named after state 10: it resets all the flags and the robot goes to state 0, ready for a new interaction

Libraries

The robot runs all of its logic on an Arduino board, so we implemented the code with Arduino IDE. We decided to use a modular approach, thanks to the **StateMachine.h** library that allows us to define the code as a state-machine diagram and lets us implement the transition between states.
To accomplish this task we used a bunch of libraries:

- **LedControl.h:** manages the eyes of Hato (i.e., led matrices) with some animations of the leds of the matrixes,
- **Servo.h:** manages the movement of the tentacles of the robot (i.e., servo motors) to let it show emotions,
- **Arduino.h:** standard Arduino libraries used for some standard things, like analog and digital I/O

- **SoftwareSerial.h:** used to manage the serial monitor, useful for calibrations and testing during the programming phase
- **DFRobotDFPlayerMini.h:** used to manage Hato's speeches, first loaded into a micro SD and then played by means of the MP3 player