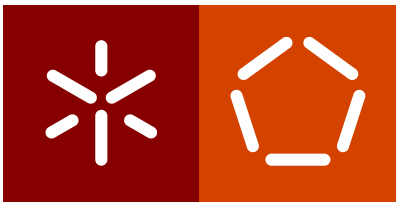

University of Minho
School of Engineering
Electronics Engineering department
Embedded systems

Project: Report

Marketing Digital Outdoor with gesture interaction — Analysis

Group 8

José Pires	A50178
Hugo Freitas	A88258

Supervised by:
Professor Tiago Gomes
Professor Ricardo Roriz
Professor Sérgio Pereira

December 13, 2021

Contents

List of Figures

List of Tables

List of Listings

List of Abbreviations

Notation	Description	First used on page nr.
CSI	Camera Serial Interface	4
DC	Direct Current	2
GPIO	General Purpose Input/Output	1
HDMI	High-Definition Multimedia Interface	1
HW	Hardware	1
mA	milliampere	2
MDO-L	MDO Local System	6
MDO-RC	MDO Remote Client	7
MDO-RS	MDO Remote Server	7
PoE	Power over Ethernet	2
SoC	System-on-a-Chip	1
SW	Software	7
TTL	Transistor-Transistor Logic	2

1. Design

In this section the theoretical foundations are used to design a viable solution, accordingly to the requirements and constraints listed. In the design phase, the product development starts, specifying the system in terms of hardware and software and its associated interfaces, the error handling required, and the design verification.

1.1. Hardware specification

The first step for system design is the HW specification. This can be pictured as a block diagram, this block diagram was already shown and mentioned in section **??** on Fig. **??**.

1.1.1. Architecture

Although that in the analysis phase an overview of the HW architecture was conceptualized, in this section, a more specific HW architecture is illustrated, using a block-diagram.

As it can be seen in Fig. 1.1, the **Raspberry Pi** is the main controller and it is powered by the **Power Supply** through **USB-C**. The **Power Supply** also supplies via **Micro-USB** the **LCD Display** control board and the **Fragrance Diffusion Actuator**. The **LCD Display** board connects to its board through **50-pin Transistor-Transistor Logic (TTL)** and the board connects to the **Raspberry Pi** through **High-Definition Multimedia Interface (HDMI)**. The **Speakers** also connect to the **LCD Display** board through **JST PH2.0**. Beyond being powered up, the **Fragrance Diffusion Actuator** also connects to the **Raspberry Pi** through the **General Purpose Input/Output (GPIO)**. What also connects to the **Raspberry Pi** through **GPIO** are the **Motion Detection** sensors. Lastly, the **Camera** connects through **Camera Serial Interface (CSI)** to the **Raspberry Pi**.

1.1.2. Main Controller

The main controller was also previously mentioned because it makes part of one of the requirements of this project: use the Raspberry Pi 4B (Fig. 1.2). This System-on-a-Chip (SoC) has several of specifications [1]:

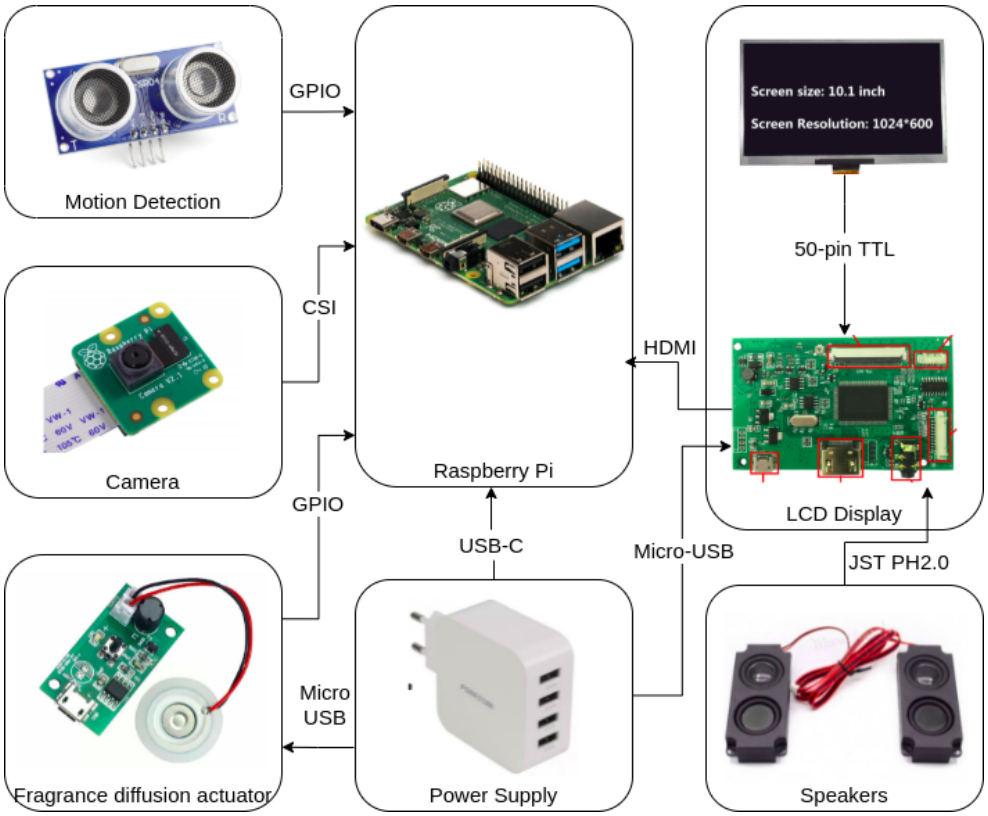

Figure 1.1.: HW architecture Block Diagram

- Processor: it has the Broadcom BCM2711 processor, quad-core Cortex-A72 (ARM v8) 64-bit with 1.5GHz;
- Memory: this model has 4GB LPDDR4 with on-die ECC;
- Connectivity: it has a 2.4 GHz and 5.0 GHz IEEE 802.11b/g/n/ac wireless LAN, Bluetooth 5.0 with low energy, one Gigabit Ethernet port, four USB ports in which two are 3.0 and another two are 2.0;
- GPIO: it has a standard 40-pin GPIO header that is fully backwards-compatible with previous boards;
- Video and Sound: it has two HDMI ports that support up to 4kp60, a 2-lane MIPI DSI display port, a 2-lane MIPI CSI camera port and a 4-pole stereo audio and composite video port;
- Multimedia: H.265 (4Kp60 decode), H.264(1080p60 decode and 1080p30 encode) and OpenGL ES 3.0 graphics;
- SD card support: Micro SD card slot for loading operating system and data storage;
- Input power: it has 5V Direct Current (DC) via USB-C connector (minimum 3A), a 5V DC via GPIO header (minimum 3A) and Power over Ethernet (PoE) - enabled (requires separate PoE HAT);
- Environment: it has a range of operation between 0°C and 50°C.

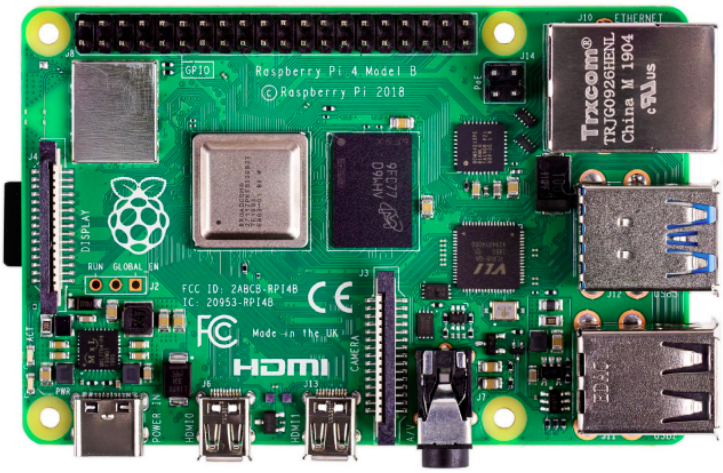

Figure 1.2.: Raspberry Pi model 4B

SD Card

Since this Raspberry supports SD Card, it will be used a micro SD Card with 16 GB that will store everything that is necessary to run the system and handle it.

1.1.3. Motion Detection

For the motion detection, it was already mentioned in section **??** that the best option is to use an ultrasonic sensor. Thus, the sensor that has been chosen is the **HC-SR04 Ultrasonic Sensor**. This sensor has the following specifications [2]:

- Operating Voltage: 5V DC;
- Operating Current: 15 milliampere (mA);
- Operating Frequency: 40 KHz;
- Maximum Range: 4 meters;
- Minimum Range: 2 centimeters;
- Ranging Accuracy: 3 millimeters;
- Measuring Angle: 15 degrees;
- Trigger Input Signal: 10 microseconds TTL pulse;
- Dimension: 45 x 20 x 15 millimeters.

Sensor Pinout

In Fig. 1.3 is described the sensor pinout and each pin works as follows [2]:

1. is the power supply for HC-SR04 Ultrasonic distance sensor which we connect to a 5V supply (for example, 5V pin on Raspberry);
2. pin that is used to trigger the ultrasonic sound pulses;
3. pin that produces a pulse when the reflected signal is received. The length of the pulse is proportional to the time it took for the transmitted signal to be detected.
4. pin that should be connected to the ground (for example, GND pin of the Raspberry).

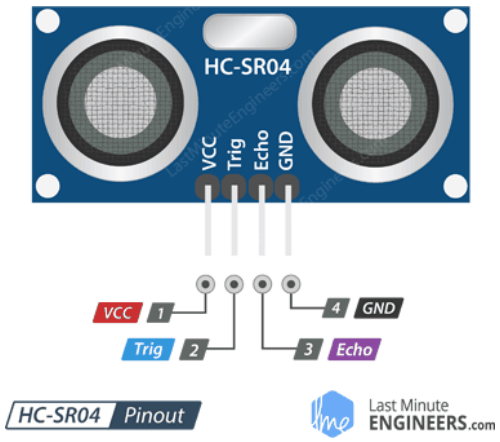

Figure 1.3.: HC-SR04 Pinout (withdrawn from [2])

For this project it will be used three sensors, one placed on the bottom, another on the middle and another one on the top of the machine. With this setup, it can be avoided some perturbations like an animal walking in front of the machine (only the bottom sensor will detect). The disposition of the middle and top sensors can't be too high, because of short people, for example.

1.1.4. Fragrance Diffusion Actuator

The chosen fragrance diffusion actuator is in Fig. 1.4 and has the following specifications [3]:

It has an operating voltage of **5V DC** and an operating current of **300 mA**. The operating power is **2 Watt**. It has a fixed frequency single-chip microcomputer with a frequency of **108 KHz**. The dimensions of the board of the module are **35 * 20 * 17 millimeters**. It has a **strong versatility**, large amount of fog, **stable performance**, the chip has an automatic timing shutdown function (4 hours of continuous work will automatically shut down protection, to turn on again, press the power on again). The 5V USB power supply mode, can be powered by MICRO charging cable. The net diameter of the atomized steel sheet is 16 millimeters, the outer diameter of the silicone ring is 20 millimeters, and the wire length is 8 centimeters.

Figure 1.4.: Fragrance module (withdrawn from [3])

1.1.5. Camera

The camera to use in this project needs to be compatible with the board in use, in this case, the Raspberry Pi. Thus the camera module that is used is the **Raspberry Pi Camera Module V2** (Fig. 1.5).

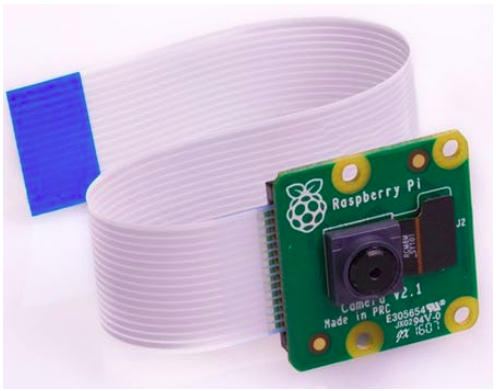

Figure 1.5.: Camera module (withdrawn from [4])

This camera module has a **Sony IMX219 8-megapixel sensor** and can be used to take high-definition video, as well as stills photographs. It supports 1080p30, 720p60 and VGA90 video modes, as well as still capture. It attaches via a 15 centimeters ribbon cable to the CSI port on the Raspberry Pi. The camera works with all models of Raspberry Pi 1, 2, 3 and 4. It can be accessed through the MMAL and V4L APIs, and there are numerous third-party libraries built for it, including the Picamera Python library. [4].

1.1.6. LCD Display

In Fig. 1.6 is the display that is used in this project. One advantage on this display is that it has **audio drivers**, which means that it is only necessary to plug a speaker and the board can handle the rest. It is also important to refer that the display isn't touch because there is no need to it and also this was the chosen one because it was the bigger and best on market considering quality and price.

As it can be seen, the display has **10.1 inches** and it is supplied with **5V DC** and with a current of **2 A** via a micro USB port. Fig. 1.7 show all the interfaces that the board module of the display provides. That was also one more reason for the choice of this display: it has an **HDMI interface** to connect to the Raspberry, the **50Pin TTL Screen Interface** that will connect to the display and two options to plug audio - the **Speaker Interface** and the **3.5mm audio interface**.

Figure 1.6.: Display (withdrawn from [5])

It can also be seen in this figure that the display has also a remote and a board to handle the remote controls, but in this implementation, it will probably not be in use.

1.1.7. Speakers

When playing video ads, it is not only necessary a display, but also a speaker to playback the sound of the ads. As it can be seen in Fig. 1.7, the screen board has two different interfaces of audio: the speaker interface and the audio interface. For this project it will be used speakers that uses the speaker interface, because that type of speakers with that interface are passive speakers and don't need a DC power supply, which is an advantage [6].

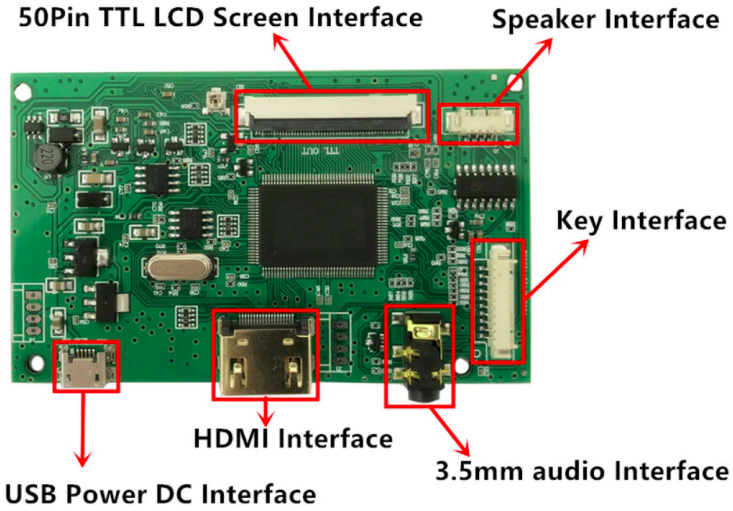

Figure 1.7.: Display Interfaces (withdrawn from [5])

Thus, the speakers that will be used have an impedance of 8 Ohm and a power of 5 Watts and are displayed on Fig 1.8.

Figure 1.8.: Speakers (withdrawn from [7])

1.1.8. Power Supply

The MDO Local System (MDO-L) will be a plugged in system, so it will be needed a plugged in power supply to supply all the components of the system. In total, the power consumption will not overtake 20 Watts and all the supplies necessary are plugged by USB (Raspberry Pi, Fragrance Diffusion Actuator, and screen). So, it will be used the power supply in Fig. 1.9, that has 4 outputs of 5 V DC and 2.4 A DC each.

Figure 1.9.: Power Supply (withdrawn from [8])

1.1.9. On/Off button

In this context, an On/Off button can be useful to power On/Off the MDO-L. However, this feature is not that necessary, so, this prototype will be only powered through the power supply previously talked in section 1.1.8 and will be always on until the power supply be unplugged, powering off the machine.

1.1.10. Total HW cost

The total HW cost can now be precisely calculated, once that all the hardware is now specified on table 1.1, yielding about 175 EUR.

Table 1.1.: Total spending on Hardware

Item	Quantity	Price (€)
Raspberry Pi 4B	1	70.00
Ultrasonic Sensor HC-SR04	3	11.70
Fragrance Diffusion Actuator	1	3.23
Raspberry Pi Camera V2	1	20.00
LCD Display	1	51.95
Speakers	1	3.00
Power Supply	1	13.50
	Total	**173.38**

1.2. Hardware interfaces definition

1.3. Software specification

Next, the SW responsible for system operation is specified for all subsystems — **MDO Remote Client (MDO-RC)**, **MDO Remote Server (MDO-RS)**, and **MDO Local System (MDO-L)**. All these subsystems are event-driven (asynchronous), and they can be more easily specified using state-machine diagrams, previously illustrated in the analysis phase (Section **??**). Also in the analysis phase, the use case diagrams helped to identify the main features required for the system and the respective sequence diagrams helped to clarify the intervening objects and the interaction among them.

In this section, the analysis phase information is used to derive the static architecture of the system — classes diagram — and to specify algorithms for its implementation through flowcharts, keeping in mind that the several subsystems operate multiple tasks concurrently, thus requiring the tasks' specification and its priorities. The data frame formats are specified for communication between the different modules. The Entities-Relationships diagram (ERD) are depicted to design the required databases and the User Interface (UI) mock-ups are recalled. The test cases for each subsystem are listed, defining its operation and the expected result. The Commercial off-the-shelf (COTS) SW and the third-party libraries are identified and a mapping between class topics and the foreseeable implementation is presented for clarification. Finally, the SW tools are listed.

1.3.1. Software architecture

The system's SW architecture was devised using Unified Modeling Language (UML) component diagrams for **Remote Client** (Fig. 1.10), **Remote Server** (Fig. 1.11), and **Local System** (Fig. 1.12). Each component diagram illustrates all SW components for the system in analysis and the interaction between them, and its interfaces with external subsystems.

Remote Client

Fig. 1.10 depicts the **Remote Client** SW architecture, encapsulated in the package **MDO-RC: AppManager**, and is comprised of the following artifacts:

- User Interface package: contains the **UI** and **UI Engine**. It is responsible for providing user feedback and capturing UI events which drive the **Remote Client**'s logic.

- Comm Manager package: manages incoming and outgoing connections to the **Remote Server** (package **MDO-RS: App Manager**), periodically checking the connection status by pinging the **Remote Server**. All connections consist of Transmission Control Protocol/Internet Protocol (TCP/IP) sockets.
- DB Manager package: manages the queries and the associated responses by building or parsing them, respectively.
- Remote Controller package: contains the **Cmd Parser**, which parses the command responses received from the **Remote Server** when the **Admin** is performing remote control of the **Local System**.
- RC Rx Parser component: high-level parser which filters between **db responses** and **cmd responses** for appropriate dispatching.
- TCP/IP Tx socket: outgoing connection node, through which **tx frames** are sent to the **Remote Server**.
- TCP/IP Rx socket: incoming connection node, through which **rx frames** are received from the **Remote Server**.

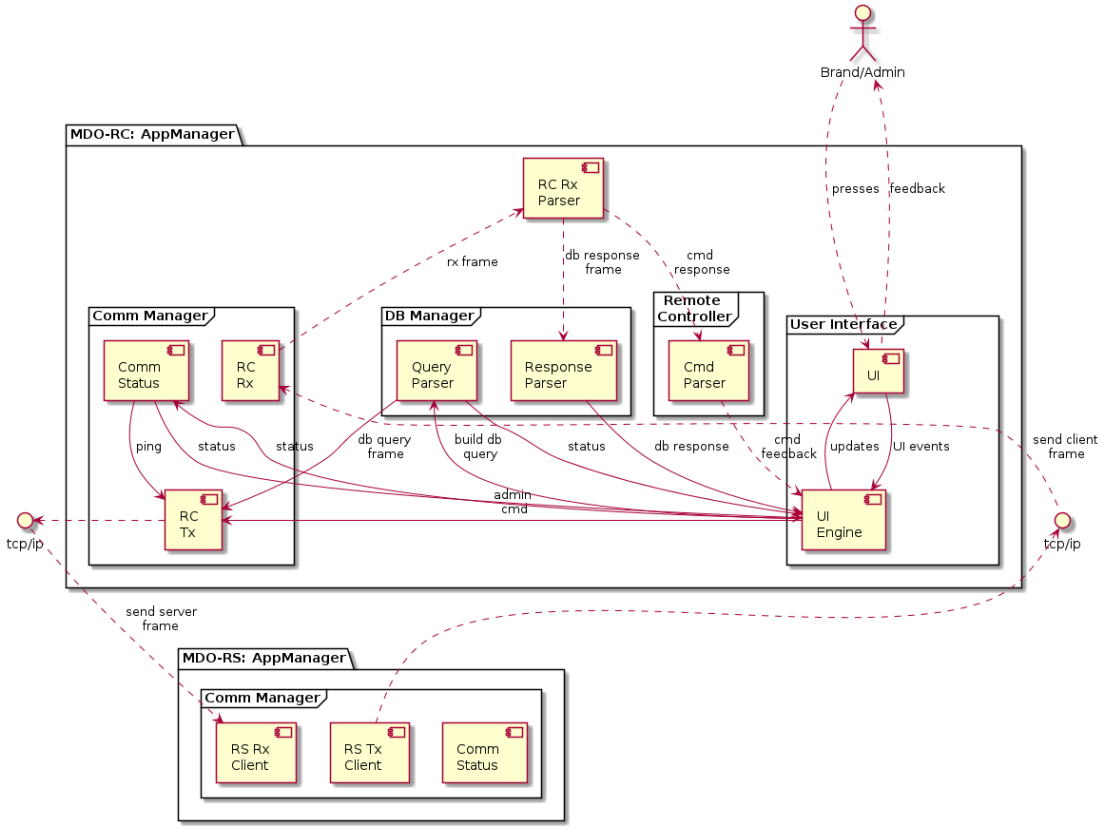

Figure 1.10.: SW architecture: component diagram — **Remote Client**

Remote Server

Fig. 1.11 depicts the **Remote Server** SW architecture, encapsulated in the package MDO-RS: AppManager. It interacts with the **Remote Client** (package **MDO-RC: AppManager**), with the **Local System** (package **MDO-L: AppManager**), and with the **Database (DB) server** (**MDO-RS: DB Server**). The database management is done using client-server architecture, with **MDO-RS: AppManager** containing the DB client and **MDO-RS: DB Server** the server.

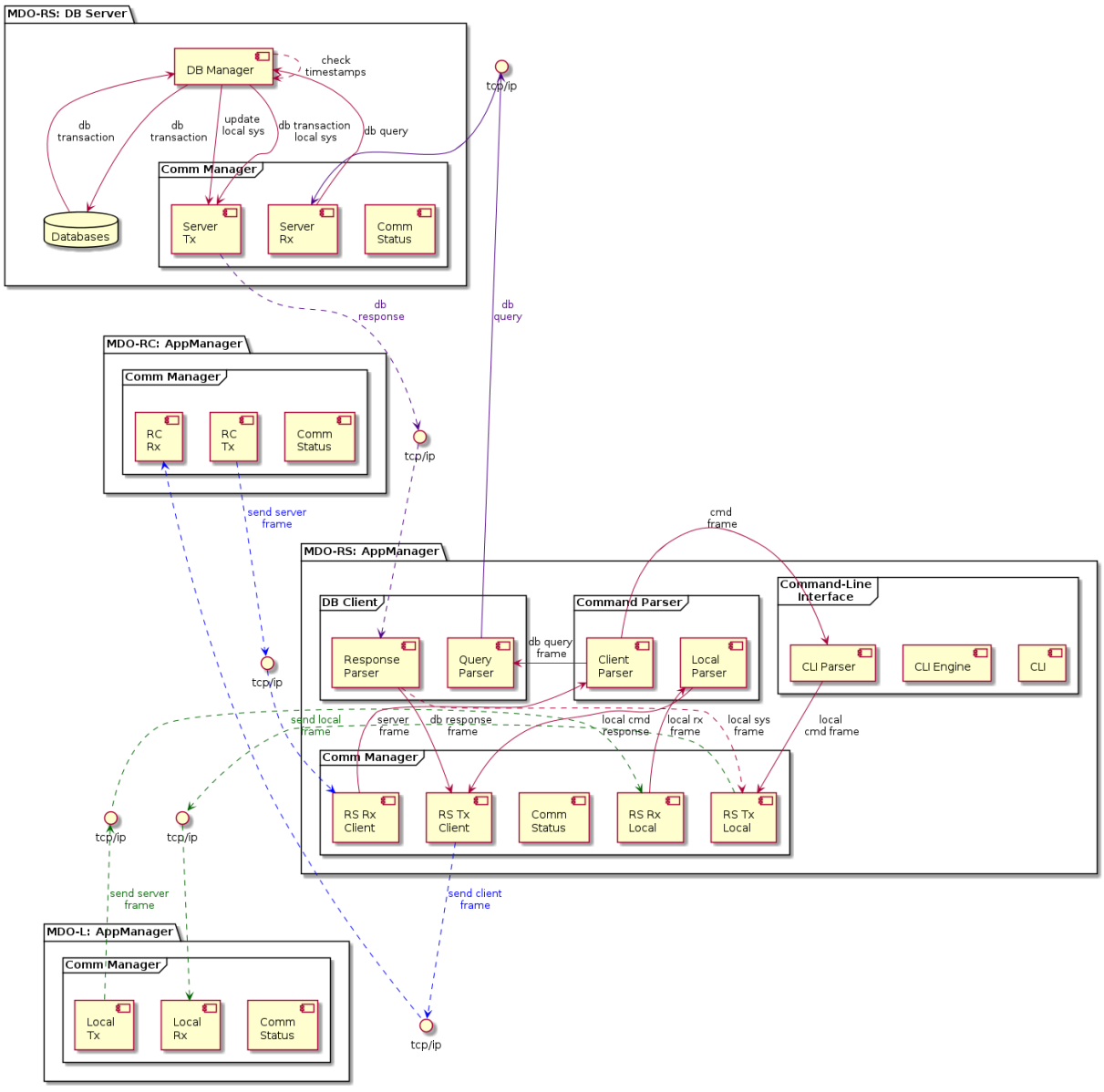

Figure 1.11.: SW architecture: component diagram — **Remote Server**

The MDO-RS: AppManager package is comprised of the following artifacts:

- Command–Line Interface package: contains the **Command Line Interface (CLI)**, the **CLI Engine**, and the **CLI Parser**. It provides the server external interface for clients to perform requests, capturing the events which drive the **Remote Server**'s logic.
- Comm Manager package: manages incoming and outgoing connections to the **Remote Client** (package **MDO-RC: App Manager**), and each **Local System** (package **MDO-L: AppManager**) it needs to interact. It periodically checks all connections statuses. All connections consist of TCP/IP sockets.
- DB Client package: manages the queries and the associated responses by building or parsing them, respectively. It performs the requests for **DB server** with the solicited queries.
- Command Parser package: contains the **Client Parser**, and the **Local Parser** to parse and handle frames received from the **Remote Client** or **Local System**, respectively, forwarding it for appropriate dispatching.
- TCP/IP sockets: incoming/outgoing connection nodes, through which incoming or outgoing traffic flows for the **Remote Client**, the **Local System** and the **DB Server**.

The MDO-RS: DB Server package is comprised of the following artifacts:

- Comm Manager package: manages incoming and outgoing connections to the **DB Client** (in package **MDO-RS: App Manager**). It periodically checks all connections statuses. All connections consist of TCP/IP sockets.
- DB Manager package: handles the received queries, issuing transactions for the databases and returns its response. It is also responsible for periodically checking timestamps, and when there is a match, update the local system with the relevant information.
- Databases: contains the actual data stored.

Local System

Fig. 1.12 depicts the **Local System** SW architecture, encapsulated in the package MDO-L: AppManager.

It interacts with:

- Remote Server (package **MDO-RS: AppManager**): to retrieve updates on its operation or **Admin** commands
- Twitter (via its **Representation State Transfer (REST) Application Programming Interfaces (APIs)**): to share posts on it
- **transfer.sh** — an Uniform Resource Locator (URL) proxy server: to ease file transfer between the **Remote Server** and the **Local System**.

The MDO-RS: AppManager package is comprised of the following artifacts:

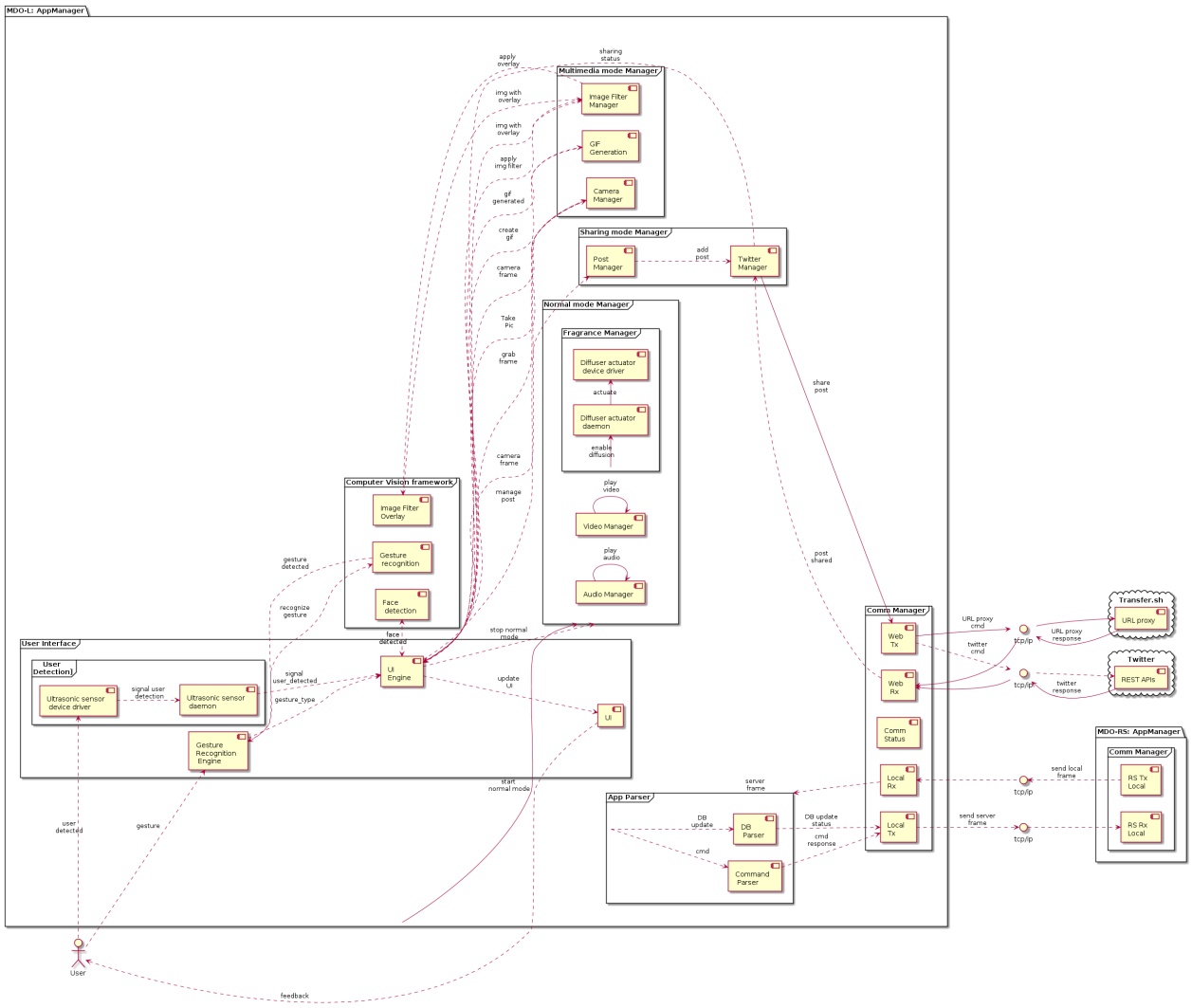

Figure 1.12.: SW architecture: component diagram — **Local system**

- User Interface package: contains the **UI**, the **UI Engine**, the **Gesture Recognition Engine**, and the **User Detection package**. The **UI** provides user feedback and **UI Engine** captures the events that drive the **Local System**'s logic. When a user approaches the **Local System**, the **Ultrasonic sensor device driver** captures this event and passes it to the user space where the **Ultrasonic sensor daemon** logs it, which, in turn, signals this event to the **UI Engine**. When the **User** performs gestures, the **Gesture Recognition Engine** requires a service to the **Gesture recognition** component (**Computer Vision Framework**) to recognize the gesture. Also, when the **User** is detected, the **UI Engine** requests the **Normal mode manager** to stop running, and requests **Face detection** to track people's faces in the camera.

- Comm Manager package: manages incoming and outgoing connections to the **Remote Server** (package **MDO-RS: App Manager**), and the internet for each web service it needs to interact (**Twitter** and **transfer.sh**). It periodically checks all connections statuses. All connections consist of TCP/IP sockets.
- Computer Vision framework package: manages the computer vision related tasks, namely, gesture recognition, face detection, and image filter overlay.
- Multimedia Mode Manager package: manages the multimedia mode related tasks, namely, image filtering, Graphics Interchange Format (GIF) generation, and camera management.
- Sharing Mode Manager package: manages the sharing mode related tasks, namely, post management, and social media management, in this case, **Twitter**.
- Normal Mode Manager package: manages the normal mode related tasks, namely, fragrance diffusion, video and audio outputs. The fragrance diffusion is requested to the device driver using a daemon to bridge user-space and kernel-space.
- App Parser package: manages the parsing for database queries and requested commands.
- TCP/IP sockets: incoming/outgoing connection nodes, through which incoming or outgoing traffic flows for the **Remote Server**, and the web services for **Twitter** and the **URL proxy server**.

1.3.2. Static architecture — Class diagrams

In this section the static architecture is derived from the SW architecture, i.e., the class diagrams are outlined, using the component diagrams as a starting point, for each subsystem.

Remote Client

Remote Server

Local System

1.4. Software interfaces definition

1.5. Start-up/shutdown process specification

1.6. Error handling specification

Bibliography

[1] Raspberry pi 4 tech specs. URL https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/. accessed: 2021-12-11.

[2] How hc-sr04 ultrasonic sensor works & interface it with arduino. URL https://lastminuteengineers.com/arduino-sr04-ultrasonic-sensor-tutorial/. accessed: 2021-12-11.

[3] Micro usb ultrasonic atomizing humidifier module fogger mist generator uk. URL https://www.ebay.co.uk/itm/203635455945?hash=item2f699e77c9:g:j-cAAOSwE75hVti0. accessed: 2021-12-11.

[4] Raspberry pi camera module 2. URL https://www.raspberrypi.com/products/camera-module-v2/. accessed: 2021-12-11.

[5] 10.1 inch hdmi screen lcd display with audio driver board monitor for raspberry pi banana/orange pi mini computer. URL https://www.aliexpress.com/item/33005274109.html?spm=a2g0o.detail.1000013.3.38072ea5MrfWq9&gps-id=pcDetailBottomMoreThisSeller&scm=1007.13339.169870.0&scm_id=1007.13339.169870.0&scm-url=1007.13339.169870.0&pvid=e1784609-f2f3-4547-9d9e-d5461fddc4c8&_t=gps-id:pcDetailBottomMoreThisSeller,scm-url:1007.13339.169870.0,pvid:e1784609-f2f3-4547-9d9e-d5461fddc4c8,tpp_buckets:668%232846%238110%231995&&pdp_ext_f=%7B"sceneId":"3339","sku_id":"12000023440173741"%7D. accessed: 2021-12-11.

[6] Passive vs. active speakers, which is right for you? URL https://www.aperionaudio.com/blogs/aperion-audio-blog/passive-vs-active-speakers. accessed: 2021-12-11.

[7] 5w speaker - para display (h). URL https://www.botnroll.com/pt/colunas-sirenes/2726-5w-speaker-para-display-h.html. accessed: 2021-12-11.

[8] Carregador 4x usb 5v / 2,4a (branco) - fonestar. URL https://www.castroelectronica.pt/product/carregador-4x-usb-5v-24a-branco--fonestar. accessed: 2021-12-11.

Appendices

A. Project Planning — Gantt diagram

In Fig. A.1 is illustrated the Gantt chart for the project, containing the tasks' descriptions.

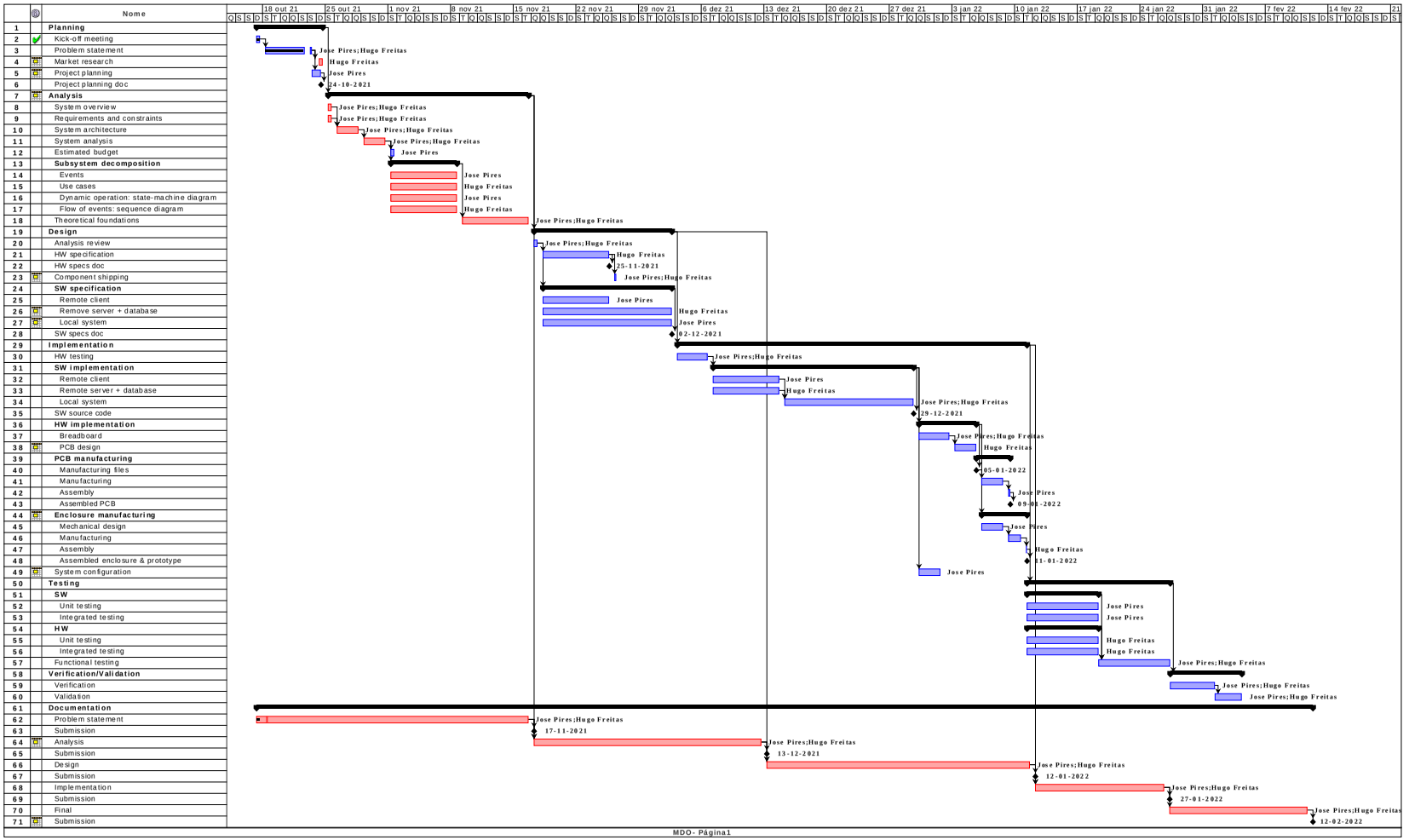

Figure A.1.: Project planning — Gantt diagram