CEA Comité Español de Automática
XLVII JORNADAS DE AUTOMÁTICA CÓRDOBA 2026
UNIVERSIDADE DA CORUÑA

Jornadas de Automática

# Integrating YARP and ROS 2 for web-based and embedded robot GUIs

Bartek Łukawski*, Grigor Hovhannisyan, Lucía M. Reques, Iñaki Ureta, Juan G. Victores, Alberto Jardón

*RoboticsLab, Department of Systems Engineering and Automation, Universidad Carlos III de Madrid, Avda. Universidad, 30, 28911, Leganés, Spain.*

**To cite this article:** Łukawski, B., Hovhannisyan, G., Reques, L. M., Ureta, I., Victores, J. G., Jardón, A. 2026. Integrating YARP and ROS 2 for web-based and embedded robot GUIs. Jornadas de Automática, 47. https://doi.org/10.17979/ja-cea.2026.47.13858

**Resumen**

En Robótica, una interfaz gráfica de usuario (GUI) permite inspeccionar el estado y configuración del robot y enviar comandos de movimiento, entre otras funciones. Pretendemos reflejar estas características en las arquitecturas software propuestas adoptando librerías de diseño de GUI (React con Vite y Blender, TouchGFXDesigner) y middlewares robóticos (ROS 2, YARP). Para su aplicación práctica, se han empleado el cobot ABB CRB 15000-5 "GoFa" y el robot humanoide TEO. En el primer caso, hemos usado ROS 2 para comunicar el robot mediante un nodo de control que envuelve la interfaz ABB Externally Guided Motion (EGM), introduciendo una aplicación web de Node.js que conecta con la red de ROS 2 mediante WebSockets y ofrece un diseño adaptable para pantallas de diferentes tamaños y resoluciones. En el segundo caso, se ha diseñado una aplicación gráfica táctil embebida con ayuda de utilidades de STM para ejecutarse en una placa Mbed conectada a la red de YARP del robot.

*Palabras clave:* Externally Guided Motion, Interfaz gráfica de usuario, Robot colaborativo, Robot humanoide, ROS 2, YARP.

**Abstract**

A graphical user interface (GUI) in robotics aids the operator in gaining insight on the robot state and configuration, and issuing motion commands, among others. We seek to fulfill these goals in the proposed software architectures by adopting GUI design frameworks (React with Vite and Blender, TouchGFXDesigner) and robotic middlewares (ROS 2, YARP). For practical demonstration, the ABB CRB 15000-5 "GoFa" cobot and the humanoid robot TEO have been considered. For the former, we leverage ROS 2 to communicate the robot through a control node wrapping the ABB Externally Guided Motion (EGM) interface, and introduce a Node.js web application that bridges the ROS 2 network via WebSockets while staying responsive to different screen sizes and resolutions of the user device. For the latter, an embedded graphical and touch-based application was designed with STM tools to run on an Mbed board connected to the YARP network of the robot.

*Keywords:* Collaborative robot, Externally Guided Motion, Graphical user interface, Humanoid robot, ROS 2, YARP.

## 1. Introduction

Manufacturers usually provide screen-based devices for interfacing with their equipment, as ABB does with its proprietary tablet-like FlexPendant. Thanks to advanced vendor tools (EGM) and widely used open frameworks (ROS 2), it is possible to create custom applications that mimic the FlexPendant, focusing on selected functionalities, yet providing additional features in accordance with the capabilities of the device. We seek to replace or complement said tablet with other handheld devices such as mobile phones.

This work also builds on our previous experiences developing user interfaces for the TEO humanoid robot. Motivated by its ongoing electronics-related refurbishments, a tactile screen has been deemed an interesting addition to provide a quick and intuitive means for accessing basic robot state and motion control, without the need of starting a session on either the robot's onboard PCs or the external computers linked to it through a local network. The proposed interface described later was inspired by the "yarpmotorgui" desktop graphical application for PC bundled with YARP.

*Corresponding author: blukawsk@ing.uc3m.es

This document is organized as follows. Section 2 expands on previous and related works. Section 3 lists the required materials and tools. The proposed architecture is detailed in Section 4, following with its implementation in Section 5. Finally, conclusions are drawn in Section 6.

## 2. Background

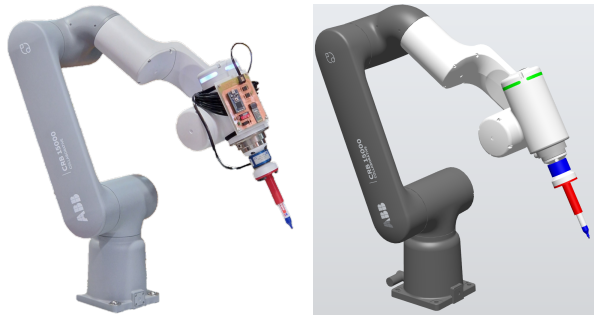
Graphical user interfaces pose an intuitive means of accessing robots, seeing an increased adoption in recent years paired with the availability of open frameworks and tools. The ROS middleware enables such implementations to achieve specific goals. For instance, some applications include simple teleoperation interfaces with sliders for educational purposes (Gomez et al., 2016), or visual aids for localization, mapping and object identification in mobile robots implementing specialized technologies such as WebGL to bridge the Gazebo simulator (Rajapaksha et al., 2021). In other contexts, humanoids such as Zeno have been used for interaction with children through a custom interface, exhibiting positive feedback (Banthia et al., 2016).

The present work is based on previous developments within the RoboticsLab team. The AppStudio editor and OmniCore App SDK had been used to implement custom FlexPendant applications for the "GoFa" robot (Łukawski et al., 2025a). On the other hand, TEO gained a React-based web application with sliders, while also accessing additional components such as cameras, force-torque sensors, or inertial sensors (Łukawski et al., 2025c). Finally, a previous practical study on the EGM interface paved the way for the architecture presented here (Łukawski et al., 2025b).

## 3. Materials and tools

### 3.1. ABB CRB 15000-5 "GoFa"

Figure 1 depicts the ABB CRB 15000-5 "GoFa" collaborative manipulator arm, both the real hardware and its simulated model in ABB RobotStudio, having six degrees of freedom (DoF) and a custom pen tool attached to its end effector. This robot features a safety controller that constantly monitors several motion parameters in runtime (such as torques exerted on each joint) to automatically act upon collisions or excess force applied onto the environment.



(a) Real robot (b) RobotStudio model

Figure 1: ABB CRB 15000-5 "GoFa".

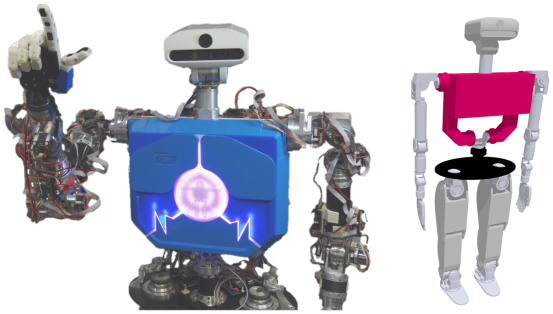
### 3.2. ABB Externally Guided Motion and ROS 2

The proposed architecture regarding this robot relies on the Externally Guided Motion (EGM) feature of modern ABB controllers, which enables high frequency (250 Hz) control loops for command processing and state feedback. This facilitates the generation of custom trajectories in the joint and Cartesian spaces, along with real-time representation of the current robot's configuration, effectively moving the control layer out of the robot to an external processing computer.

The ROS 2 middleware was leveraged to introduce a custom "abb_egm_driver" node wrapping the EGM communications layer.[1] Besides motion and feedback, this node also allows setting a binary signal on the robot side, and sharing an array of up to forty double precision floats between the robot and the external controller. For the purpose of achieving point-to-point movements using the proposed slider interface, a trajectory generator was implemented in said node to apply rectangular or trapezoidal velocity profiles during execution.

### 3.3. TEO humanoid robot

The second robot platform, the 28-DoF full-sized humanoid TEO developed by the RoboticsLab research team at Universidad Carlos III de Madrid, is represented in Figure 2. Beyond manipulation and locomotion tasks, this robot features a range of sensing and peripheral devices such as RBG and RGBD cameras, inertial sensors and force-torque sensors, as well as a speaker and a microphone for human-robot interaction. At the time of writing this document, it is undergoing a ground-up refurbishment of its power supply electronics, which will be followed by further improvements in the connectivity and mechanical areas. The addition of the Mbed touchscreen is framed within these efforts.



(a) Real robot (b) Gazebo model

Figure 2: TEO humanoid robot by RoboticsLab UC3M.

### 3.4. Mbed DISCO-F746NG

Due to its tactile capabilities, the Mbed DISCO-F746NG development board was devised for installation on the back of TEO, with a second board acting as a backup when connected to an external PC on the same local network. This board features an STM 32-bit Arm Cortex-M7 chip with several network interfaces: UART, I2C, SPI, CAN, Ethernet. Serial port through UART/USB has been chosen due to its bandwidth and existing software wrappers. The TouchGFXDesigner software framework was used to develop graphical applications.

[1] https://github.com/roboticslab-uc3m/abb_egm_driver

## 4. Architecture

### 4.1. ROS 2 via WebSockets and Node.js module with React

The proposed architecture to teleoperate the GoFa robot through a graphical user interface (GUI) is depicted in Figure 3. An interface is established with the robot controller, either real or virtual, through an EGM client-server pair wrapped by the custom "abb_egm_driver" ROS 2 node described earlier. The ROS 2 middleware enables an abstraction layer between components communicated through ports (or "topics" in ROS slang) (Quigley et al., 2009).

The GUI was developed using the React framework and communicated with the ROS 2 node via WebSockets using the roslibjs library[2], then managed by Node.js to be run as a web server application served by HTTP with JavaScript/CSS over the local network (LAN). The rosbridge_server node[3] relays all WebSockets requests through the ROS 2 network.

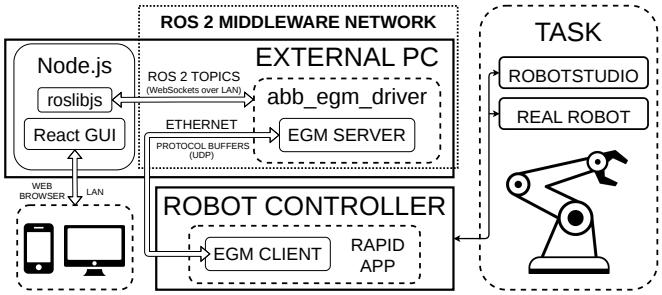



Figure 3: Proposed architecture of the graphical application on the "GoFa".

### 4.2. YARP-based Mbed connector via serial port

On TEO, a different architecture is proposed as shown in Figure 4. It is based on the YARP middleware and robotics framework, which shares similarities with ROS 2 regarding its purpose and functionality (Metta et al., 2006). Compatibility with ROS 2 is still possible to introduce high-level applications, such as teleoperation-oriented tasks, interfacing with the robot through a Cartesian controller (Łukawski et al., 2025c).

On the lower level, joint controllers are closest to the real robot or its virtual counterpart modeled in Gazebo Sim or a similar simulator. A connector application bridges the joint controllers through YARP ports with the embedded GUI application executing on the Mbed device, connected via serial port (USB). YARP enables the Mbed device to be located anywhere as long as it belongs to the same LAN as the robot controllers: mounted on TEO's back and connected to its computers aboard, or connected to an external control PC.

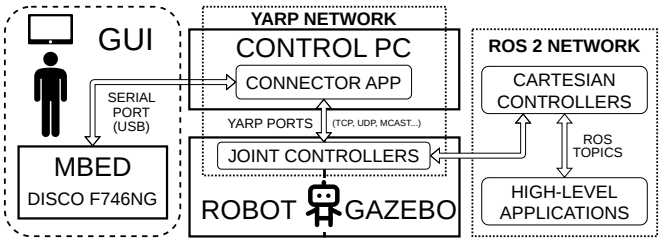


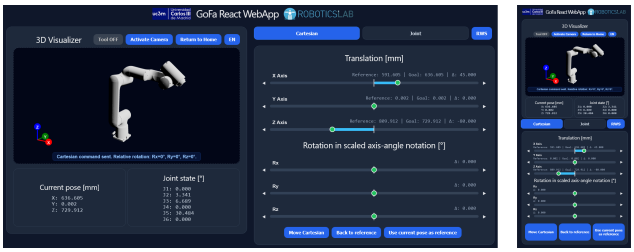

Figure 4: Proposed architecture of the graphical application on TEO.

[2] https://github.com/RobotWebTools/roslibjs
[3] https://github.com/RobotWebTools/rosbridge_suite

## 5. Implementation

### 5.1. React-based web application

The frontend layer of the user interface was implemented in JavaScript/TypeScript using React and Vite, while the backend runs on Node.js to serve the web application over the network. Code of a previous project was reused in which an equivalent web application was developed for the TEO robot (Łukawski et al., 2025c). Similarly, this design features a command interface based on sliders, as depicted in Figure 5.




(a) Desktop (b) Mobile

Figure 5: React-based web interface on different device types.

Two sets of sliders have been exposed to command the robot depending on the space: joint or Cartesian commands. In joint command mode, there are as many sliders as robot axes (six). In Cartesian command mode, sliders are grouped by threes: translation along XYZ axes, and rotation expressed in scaled axis-angle notation. Each slider features three indicators: a green circle indicating the next target, a smaller orange circle indicating the current state, and a vertical pipe marker corresponding to the stored reference value. Above each slider, three numeric values are shown: the current reference, the target, and the distance between both. Motion is started upon clicking a button below the slider group, which initiates a point-to-point movement handled by the embedded trajectory generator of the ROS 2 node. Additional buttons allow to move back to the previous reference position, and to store a new reference. Finally, at the left and right margins of each sliders, an arrow-like button initiates a continuous, incremental motion of the respective degree of freedom while pressed. This latter mode overrides the trajectory generation.

Above the slider interface, the mode selector is displayed along with a button to access some Robot Web Services (RWS) commands, including operation mode selection and motor on/off state switching. On the left side, an interactive visualization of the robot is shown, built with Blender. Forward kinematics is performed underneath to match the current robot state with this representation. It features a "ghost" clone so that the future configuration of the robot is previewed whenever the user acts upon the sliders, prior to sending the motion command. Below this section, the current robot configuration is displayed, both in Cartesian and joint spaces.

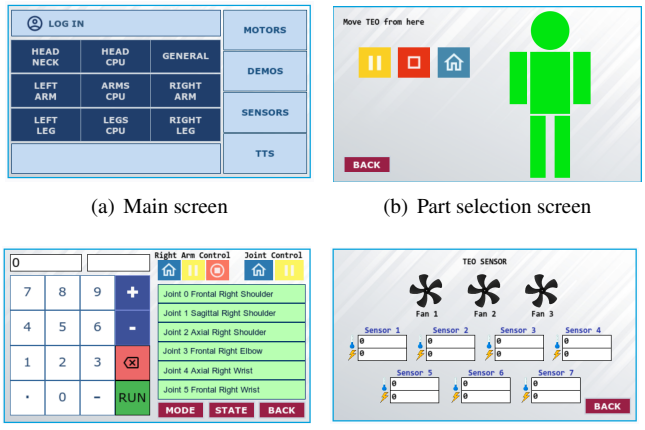
Lastly, this web application features an adaptive design that adjusts the layout to the size and width of the display. Figure 5(a) showcases the GUI as seen on a desktop device, while Figure 5(b) corresponds to a mobile device.

### 5.2. Mbed touchscreen graphical interface

The graphical application for TEO was built using STM tools that ease the development of programs and tactile user interfaces on the DISCO board, such as ToughGFXDesigner (interface design), STM32CubeIDE (backend programming), and STM32CubeMX (pin configuration). Sample screens are depicted in Figure 6. The main screen (Figure 6(a)) provides direct access to most features:

- Power switches: each of eight buttons is linked to a physical digital output pin to power on or off a CPU (head, manipulation, locomotion) or the supply (motors, drivers) of a whole limb (left/right arm, left/right leg, neck), and a ninth "general" button allows for easy disconnection of the whole robot.
- Motor control: a part selection interface is shown (Figure 6(b)) along with buttons for stopping all robot joints at once or setting them to an initial position. A more detailed per-part screen (Figure 6(c)) features a virtual keyboard for sending motor commands to individual joints. Live encoder measurements, control modes and stored axis names can be inspected here.
- Demo execution: contains shortcuts for issuing predefined motions and launching complex demonstrations.
- Sensor readings: current and temperature measurements of the respective sensors connected to the Mbed through I2C interface ports (Figure 6(d)).
- TTS interface: language (English and Spanish) and voice selectors along with an input field and a virtual keyboard for introducing the text to be transcribed.
- Logger: latest info/warning/error events, e.g. hardware faults along with their timestamp and description.

A custom serial protocol is shared between the Mbed and the Python connector. YARP is leveraged to delegate most processing (e.g. motion commands, demo execution, TTS transcription) to the PCs aboard TEO.




(a) Main screen (b) Part selection screen

(c) Motor control screen (d) Sensor readings screen

Figure 6: Mbed-based graphical user interface (GUI) for TEO.

[4] https://github.com/roboticslab-uc3m/teo-mbed-gui
[5] https://github.com/roboticslab-uc3m/gofa-react-webapp

## 6. Conclusions

In this work, two architectures are presented leveraging ROS 2 and YARP for the design of graphical user interfaces devoted to motion control and diagnosis of robot platforms. A higher-level web application was developed using React to interface with the "GoFa", whereas STM utilities were used to design an embedded application for an Mbed platform on TEO, thus achieving both direct hardware access to its sensors, and joint control commands and state feedback through serial port connection with the onboard PCs. The source code has been published on GitHub under an open license.[4][5]

Regarding future work, the React-based application could benefit from accessing the mobile device's inertial sensors for real-time teleoperation based on their measurements, as demonstrated in earlier developments for TEO.

## Acknowledgments

This research has been financed by "iRoboCity2030-CM, Robótica Inteligente para Ciudades Sostenibles" (TEC-2024/TEC-62 funded by "Programas de Actividades I+D en Tecnologías de la Comunidad de Madrid"); "ROBOASSET: Intelligent robotic systems for assessment and rehabilitation in upper limb therapies" (PID2020-113508RB-I00, financed by AEI/10.13039/501100011033); "iREHAB: AI-powered Robotic Personalized Rehabilitation" (DTS22/00105, financed by Instituto de Salud Carlos III (ISCIII)); and EU structural funds.

## References

Banthia, V., Maddahi, Y., May, M., Blakley, D., Chang, Z., Gbur, A., Tu, C., Sepehri, N., 2016. Development of a graphical user interface for a socially interactive robot: A case study evaluation, in: Information Technology, Electronics and Mobile Communication Conference (IEMCON), IEEE. doi:10.1109/IEMCON.2016.7746294.

Gomez, C., Hernandez, A., Crespo, J., Barber, R., 2016. Learning robotics through a friendly graphical user interface, in: ICERI2016 Proceedings, IATED. pp. 483–492. doi:10.21125/iceri.2016.1120.

Łukawski, B., Cervera, M.M., Pascual, C.M., Oña, E.D., Victores, J.G., Jardón, A., 2025a. Espresso macchiato, por favore: collaborative robotic coffee-making for education, in: Jornadas de Automática, CEA. doi:10.17979/ja-cea.2025.46.12274.

Łukawski, B., Oña, E.D., Jardón, A., Victores, J.G., Balaguer, C., 2025b. Development of educational applications with ABB GoFa collaborative robot using Externally Guided Motion, in: Int. Conf. on Control, Automation and Diagnosis (ICCAD), IEEE. doi:10.1109/ICCAD64771.2025.11099395.

Łukawski, B., Rebollo, M., Gilabert, Á., Victores, J.G., Balaguer, C., Jardón, A., 2025c. YARP Cartesian controller layers over ROS 2 for teleoperation and web applications, in: Jornadas de Automática, CEA. doi:10.17979/ja-cea.2025.46.12252.

Metta, G., Fitzpatrick, P., Natale, L., 2006. YARP: Yet Another Robot Platform. International Journal of Advanced Robotic Systems 3, 43–48. doi:10.5772/5761.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y., et al., 2009. ROS: an open-source Robot Operating System, in: ICRA workshop on open source software, Kobe. p. 5.

Rajapaksha, D.D., Nuhuman, M.N.M., Gunawardhana, S.D., Sivalingam, A., Hassan, M.N.M., Rajapaksha, S., Jayawardena, C., 2021. Web based user-friendly graphical interface to control robots with ROS environment, in: Int. Conf. on Information Technology Research (ICITR), IEEE. doi:10.1109/ICITR54349.2021.9657337.