The Technical Stacks Behind Botronics' iXi Autonomous Golf Trolley

Antoine Van Malleghem
Enzo Ghisoni
David Moli

FOSDEM'26

Antoine Van Malleghem

CTO & Co-Founder

Enzo Ghisoni

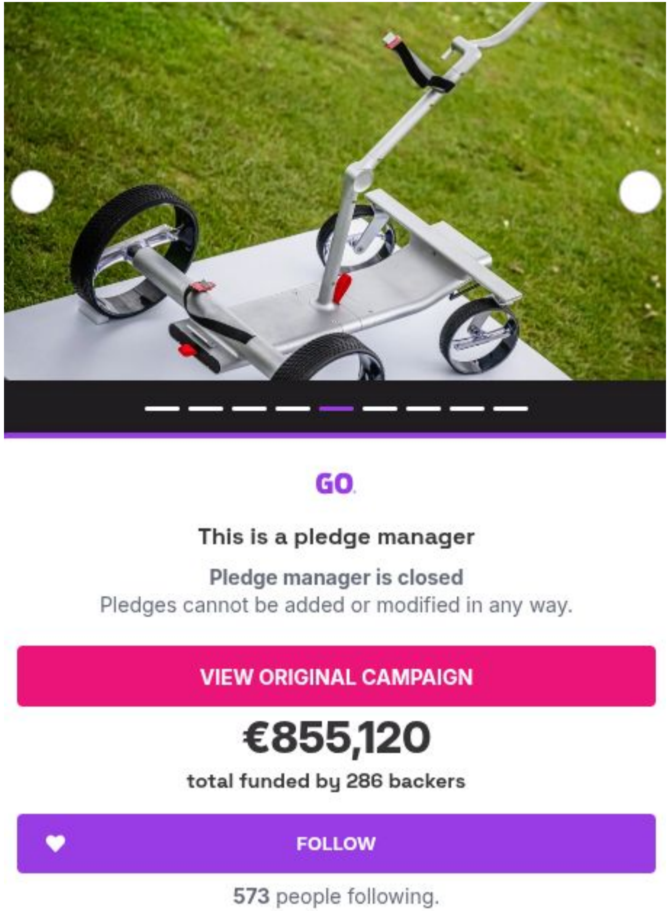
Robotics Engineer
Joined in 2024

David Moli

Robotics Engineer
Joined in 2025

Botronics

- From 2022
- Belgium (HQ in Nivelles) 🇧🇪
- Just closed a seed round of 1.6M€
- 10 people
- ❤️ Open Source
  - Friend of Nav2
  - OSRA Supporting Individual(s)
  - Contribute to open source (ROS packages, T&T parties,...)
  - ROS Meetup Belgium organizer



This is a pledge manager

Pledge manager is closed
Pledges cannot be added or modified in any way.

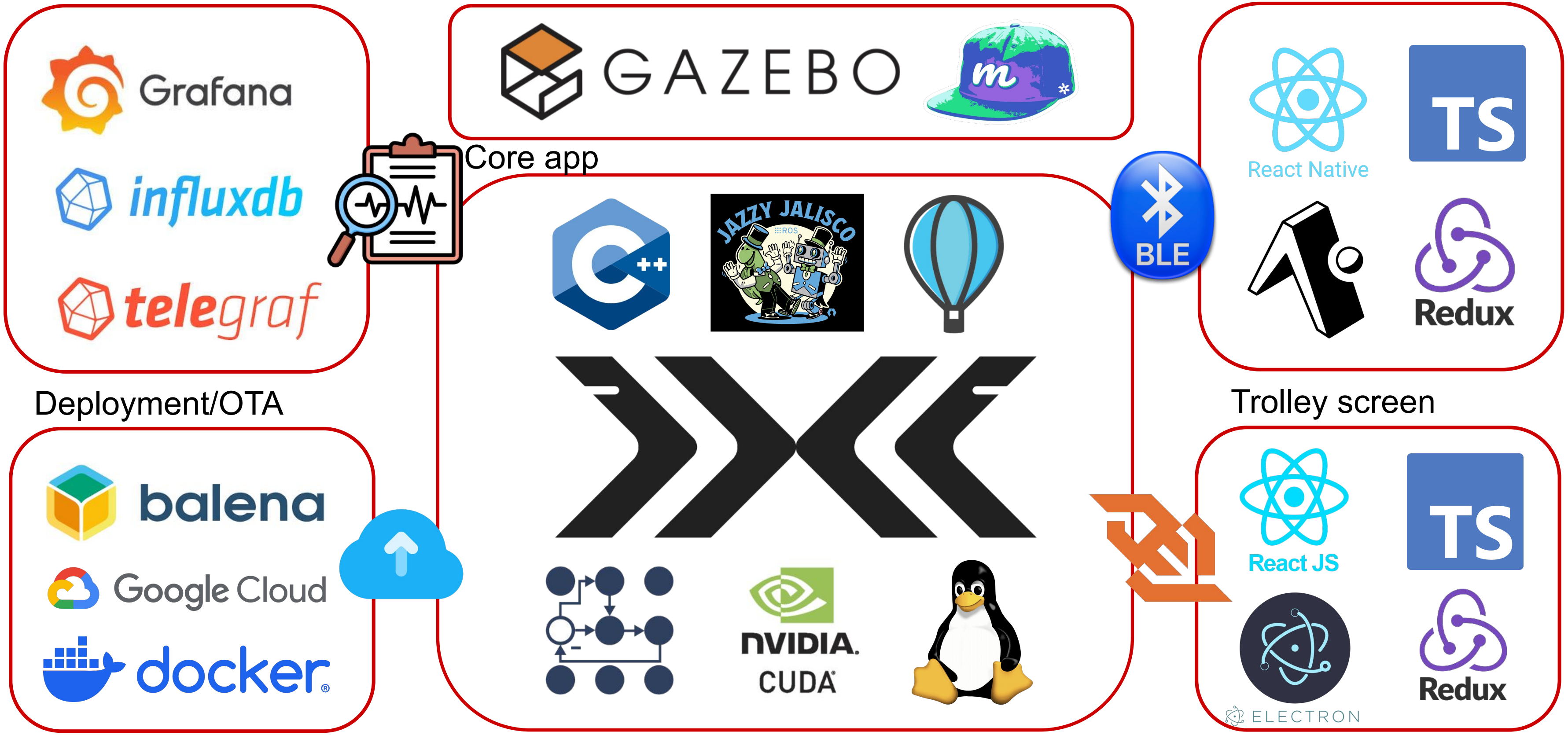
€855,120
total funded by 286 backers

573 people following.

- First ever autonomous golf trolley
- Sold ~300 units in Crowd-funding
- Final Price : ~5000€
- Currently in industrialization phase

Product constraints

- B2C product
  - Price matters more
  - Avoid initial setup
  - UX
- Outdoor robotics
  - Private environment
  - Semi known environment
  - Large areas
  - Golf trolleys have to follow rules
  - Golf courses have steep slopes



Observability

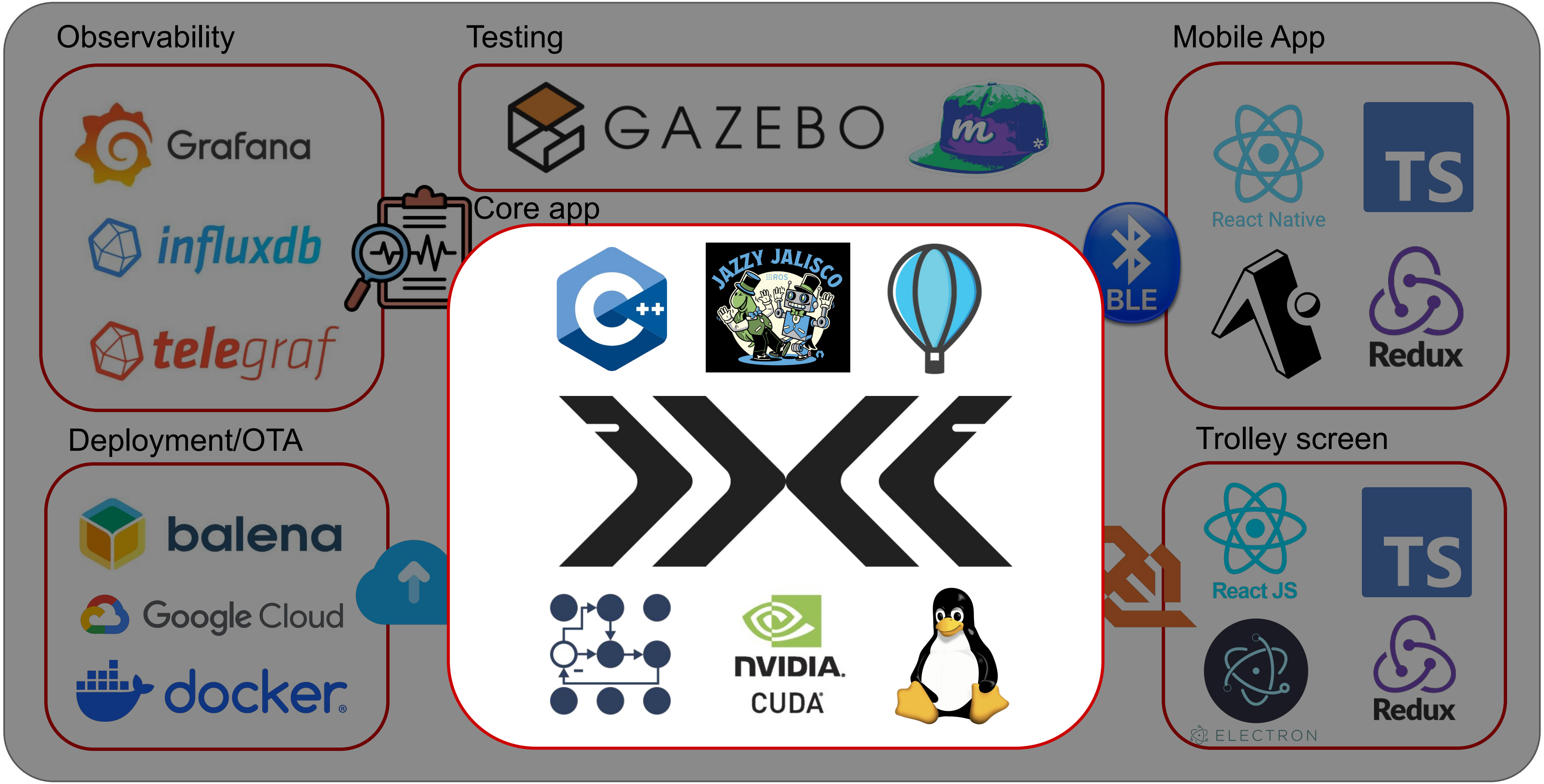
Testing

Mobile App

Core app

Deployment/OTA

Trolley screen



Observability

Testing

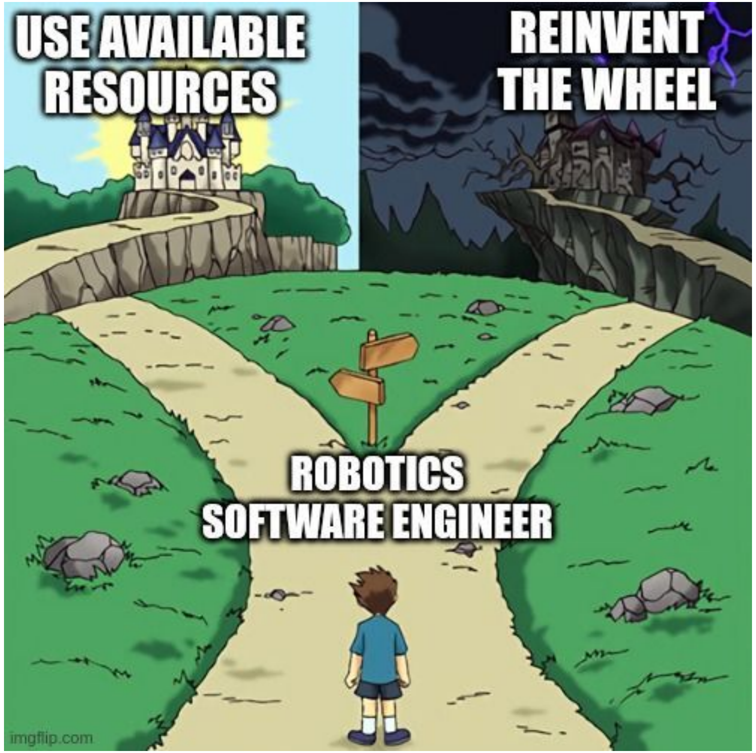
Mobile App

Core app

Deployment/OTA

Trolley screen

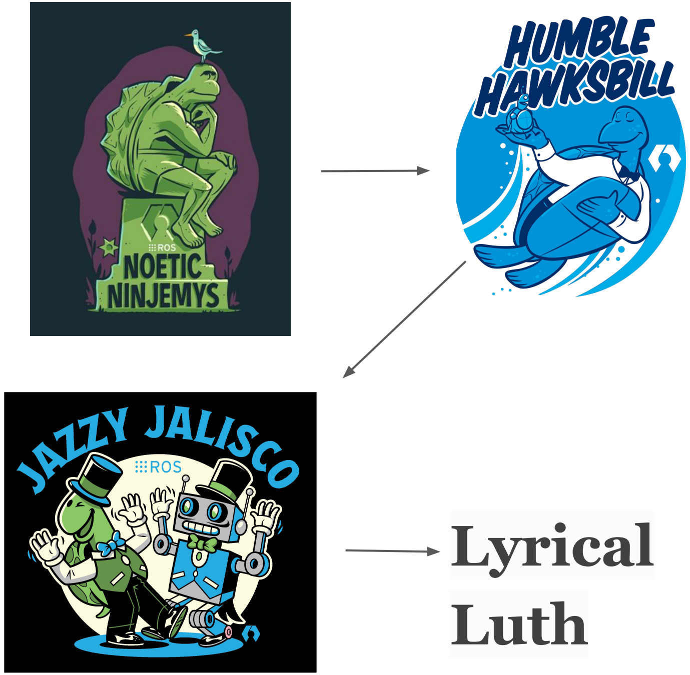
Why choose ROS 2 to develop our robot ?



- Open source 🔥
- Large and Active community
- Wide ecosystem with various packages available
- Designed for modularity and scalability

- No hard real time constraints
- No-deterministic behavior is not a big deal for us

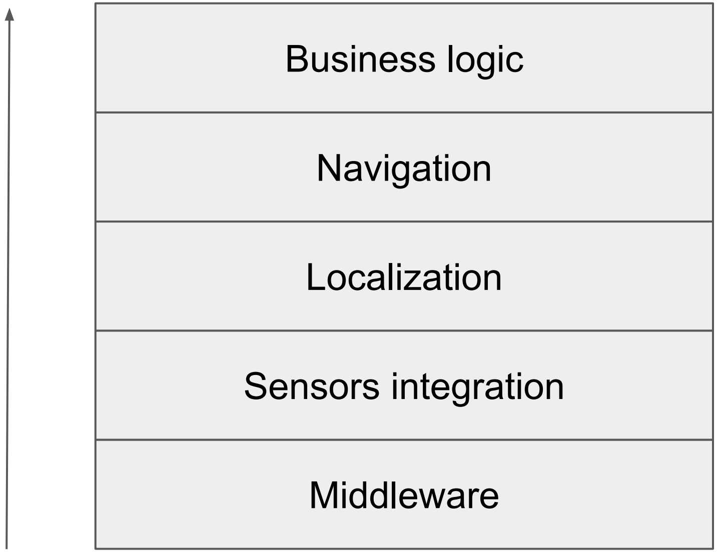
ROS 2 version and migration



Lyrical Luth

- Currently on ROS 2 Jazzy
- Advantage of frequent migration
  - Latest features from ROS 2 and packages
  - Easier for open source contribution
- For the moment we target only LTS
- Pixi to simplify the migration and be less OS dependent (cuda version) Jetpack

Why choose ROS 2 to develop our robot ?



Business logic

Navigation

Localization

Sensors integration

Middleware

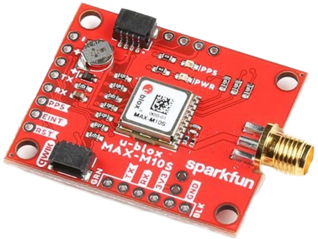
- From the ROS 2 suite
  - rclcpp, rclpy
  - ros2_control
  - ros_diagnostics
  - rmw …
- Community projects
  - Nav2
  - robot_localization
- Custom packages for Sensors drivers and Business logic …

Which RMW to choose?

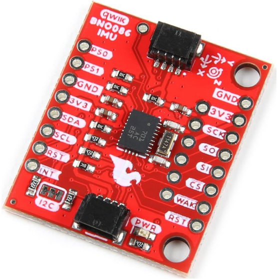
- Experience based on “localhost only” middleware
- 1 month on FastDDS
- 2 years on CycloneDDS
- 4 months and going on Zenoh
  - SHM made easy
  - 10% less CPU usage
  - Easy initial setup

Sensors

Stereo camera: SDK + Custom driver



GPS
Community driver



IMU/Magnetometer
Custom driver

ROS Guidance

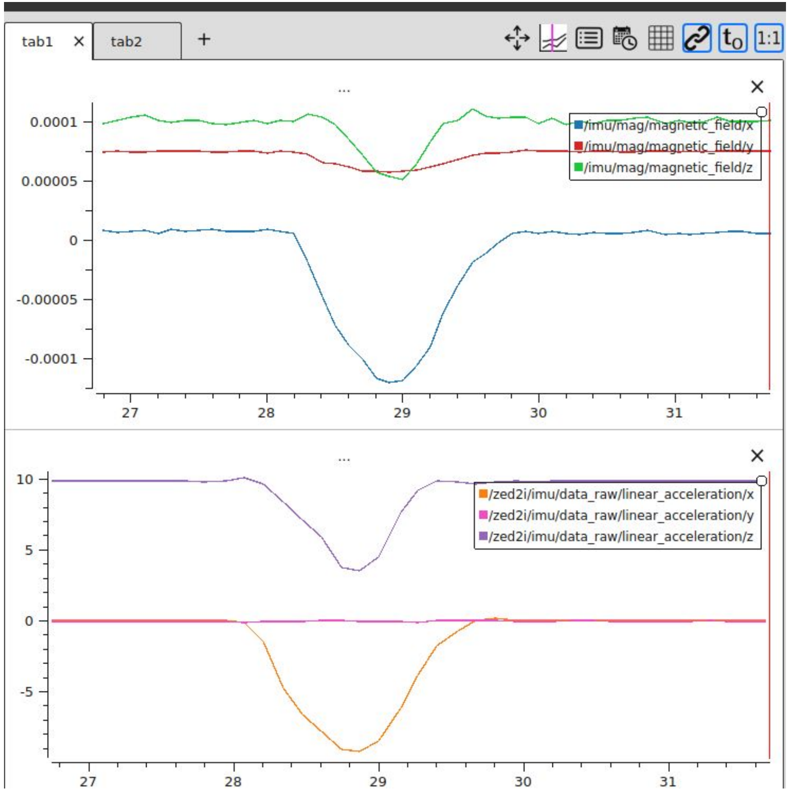
- ROS 2 can be helpful but is not enough
- If no ROS 2 driver available, follow the REPs to implement it

Not always plug and play

- Kernel configuration is sometimes needed
- We had to patch the kernel

In any case, read the datasheet and do some basic tests without ROS

Sensors


tab1 ×
tab2
+
...
0.0001
0.00005
0
-0.00005
-0.0001
27
28
29
30
31
🟦 /imu/mag/magnetic_field/x
🟥 /imu/mag/magnetic_field/y
🟩 /imu/mag/magnetic_field/z
...
10
5
0
-5
27
28
29
30
31
🟧 /zed2i/imu/data_raw/linear_acceleration/x
🟪 /zed2i/imu/data_raw/linear_acceleration/y
🟪 /zed2i/imu/data_raw/linear_acceleration/z

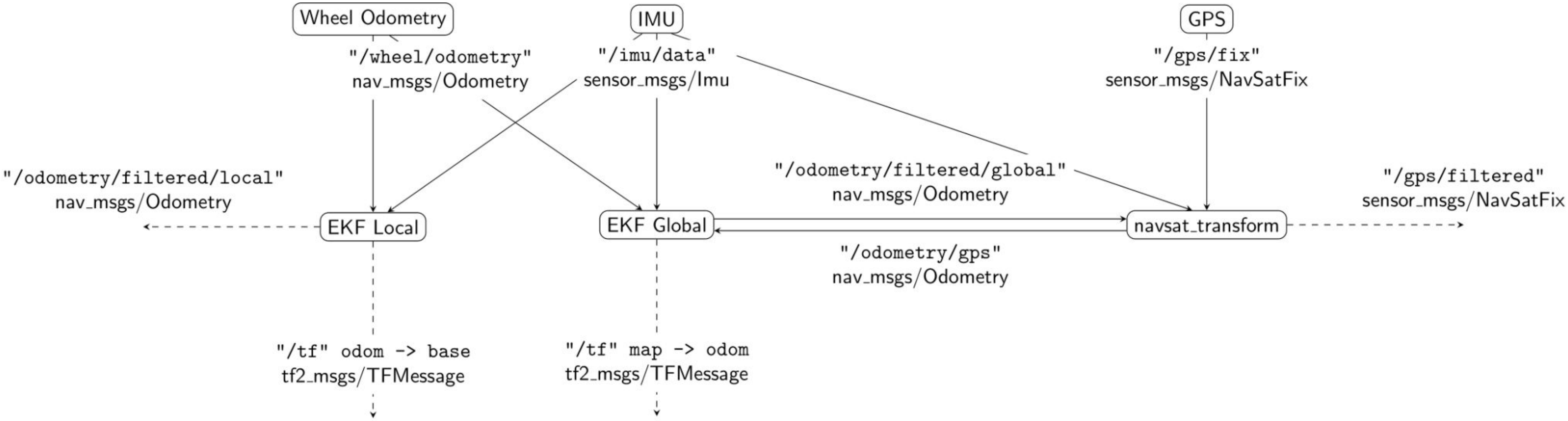
Even if you rely on a 3rd party driver:

- Verify the output data
    - IMU with or without gravity ?
    - Calibration needed ?
    - Parameters ?
    - Even headers need to be verified
- Visualize the data. We use PlotJuggler for this

Another recommendation:

- Don’t be too binded with hardware
    - Avoid all in one sensors
    - Avoid all in one SDK

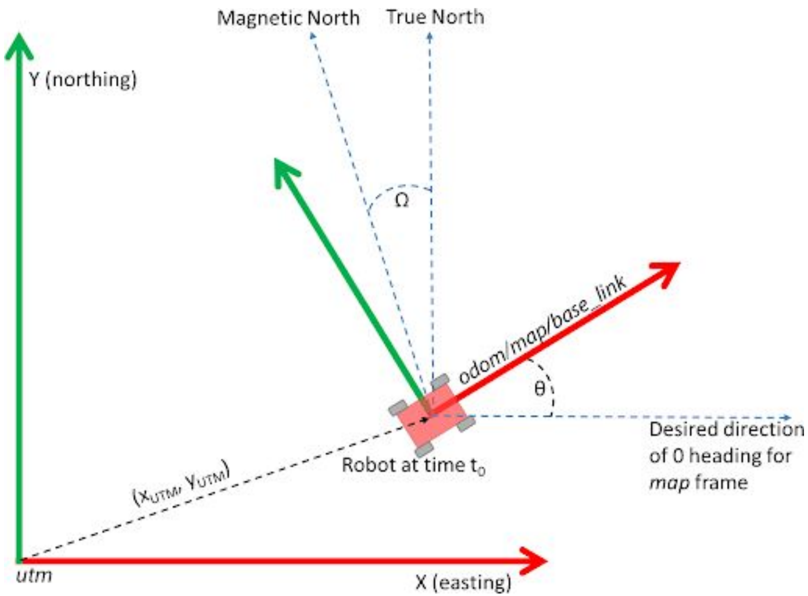
Localization



- Outdoor localization
  - Rely on GPS for global localization
  - 1st EKF for local frame (odom)
  - 2nd EKF for global frame (map)

We use the robot_localization package for sensor fusion and for navsat_transform

Localization



- Visual odometry
  - Not reliable enough on golf courses
  - No many features
- Magnetic declination is an important factor for GPS localization
  - Must work all around the world
  - Change with time
- Need to compute initial pose

Lessons learned:

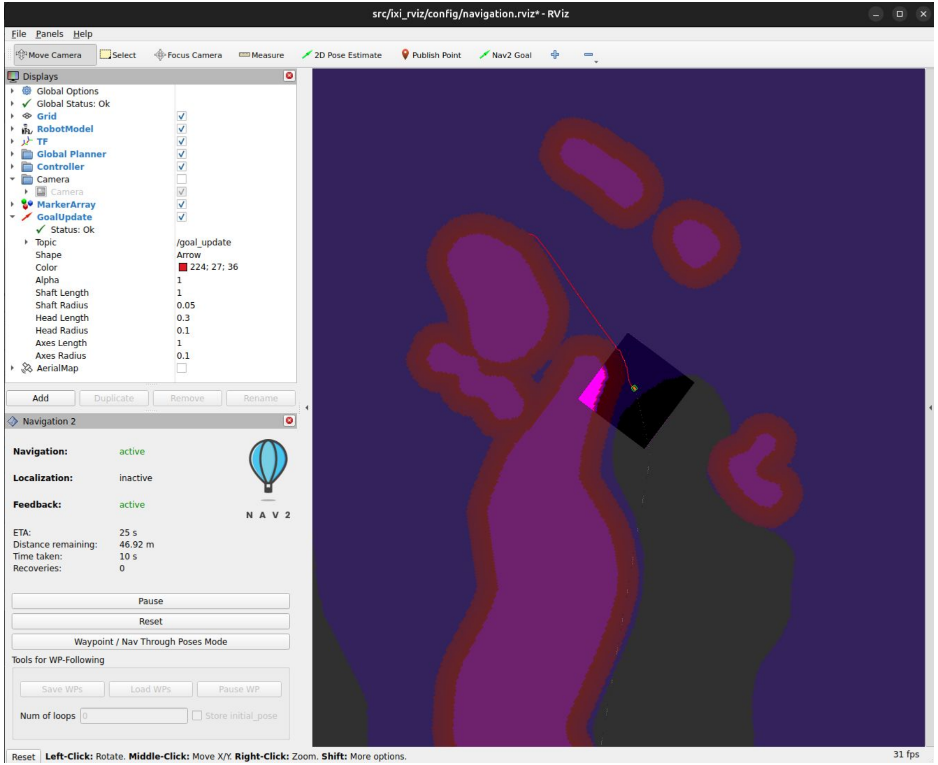
➢ Make your localization reliable before starting to dive in navigation

Nav2 for Autonomous Navigation

- Advanced features out of the box
  - Path planner
  - Controllers
  - Collision monitors…
- Highly modular architecture
  - Rely on Behavior Trees
  - Custom plugins
- Complete framework with best practices

- Well maintained by a clear structure

Our use of Nav2



Global Options	
Global Status: Ok	
Grid	☑
RobotModel	☑
TF	☑
Global Planner	☑
Controller	☑
Camera	□
Camera	☑
MarkerArray	☑
GoalUpdate	☑
Status: Ok	
Topic	/goal_update
Shape	Arrow
Color	🟥 224; 27; 36
Alpha	1
Shaft Length	1
Shaft Radius	0.05
Head Length	0.3
Head Radius	0.1
Axes Length	1
Axes Radius	0.1
AerialMap	□

Navigation: active

Localization: inactive

Feedback: active

ETA: 25 s
Distance remaining: 46.92 m
Time taken: 10 s
Recoveries: 0

Num of loops 0 □ Store initial_pose

Left-Click: Rotate. **Middle-Click:** Move X/Y. **Right-Click:** Zoom. **Shift:** More options.

- No SLAM (semi-known environment)
- Large Statics maps of Golf Courses
  - ➢ Bunkers, Ponds, Greens
  - ➢ Define GPS goal based on it
- Obstacle avoidance from stereo vision
  - ➢ Only front view from camera
  - ➢ Depth to laserscan (performances)
  - ➢ Spatio temporal voxel layer
- Custom plugins

Nav2 - Custom feature: Follow

The robot follows the user based on vision:

- Custom behavior tree
- Custom plugins
  - Behavior
  - Navigator

- Benefits
  - Easy integration with BT
  - Easy access to navigation data/toolbox out of the box
    - ➢ localization
    - ➢ costmaps

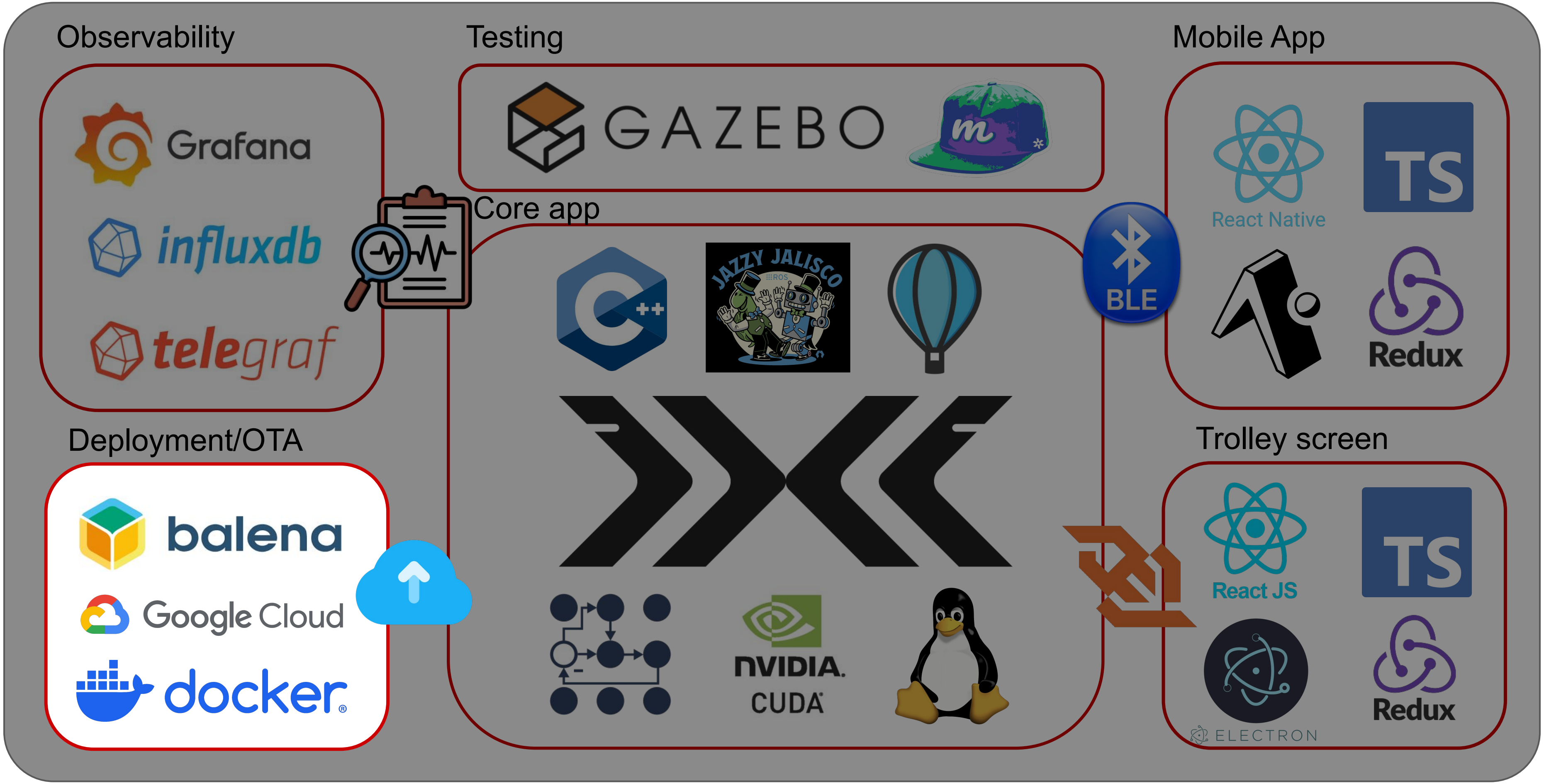
Vision Stack

Corporate needs you to find the differences between this picture and this picture.
YOLO
They're the same picture.
imgflip.com

Hardware agnostic software

- Object detection:
  - Yolo model + transfer learning

- Object tracking:
  - Attach feature detected to a specific id
  - Avoid id switches
  - Keep the target id even with obstruction
  - BoT-SORT tracking



Observability

Testing

Mobile App

Core app

Deployment/OTA

Trolley screen

Our journey with OTA update

Jetpack (NVIDIA Jetson) OTA capability

↓

Yocto + Mender

↓

Balena

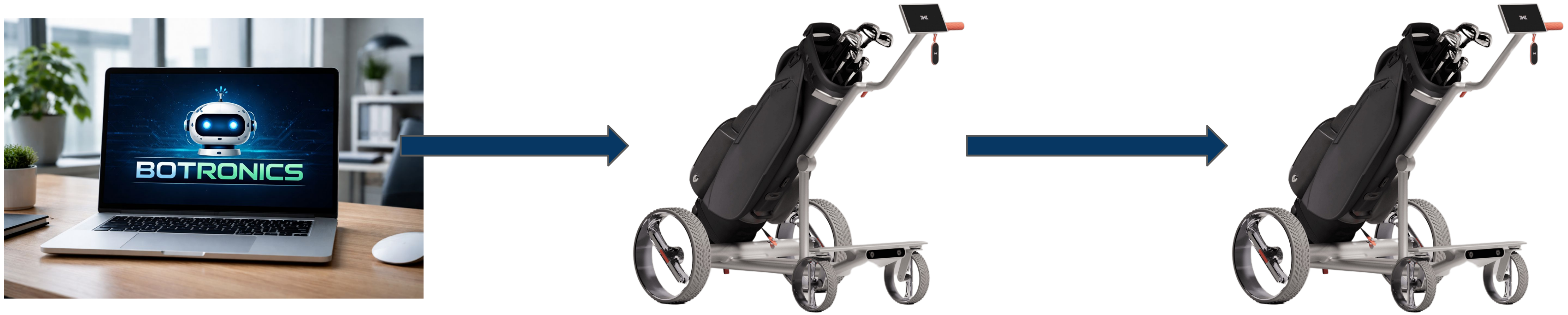
- Basic Yocto image
- Docker on top of it
- Balena Cloud

⚠️ User confirmation ⚠️

Development Flow

DEV

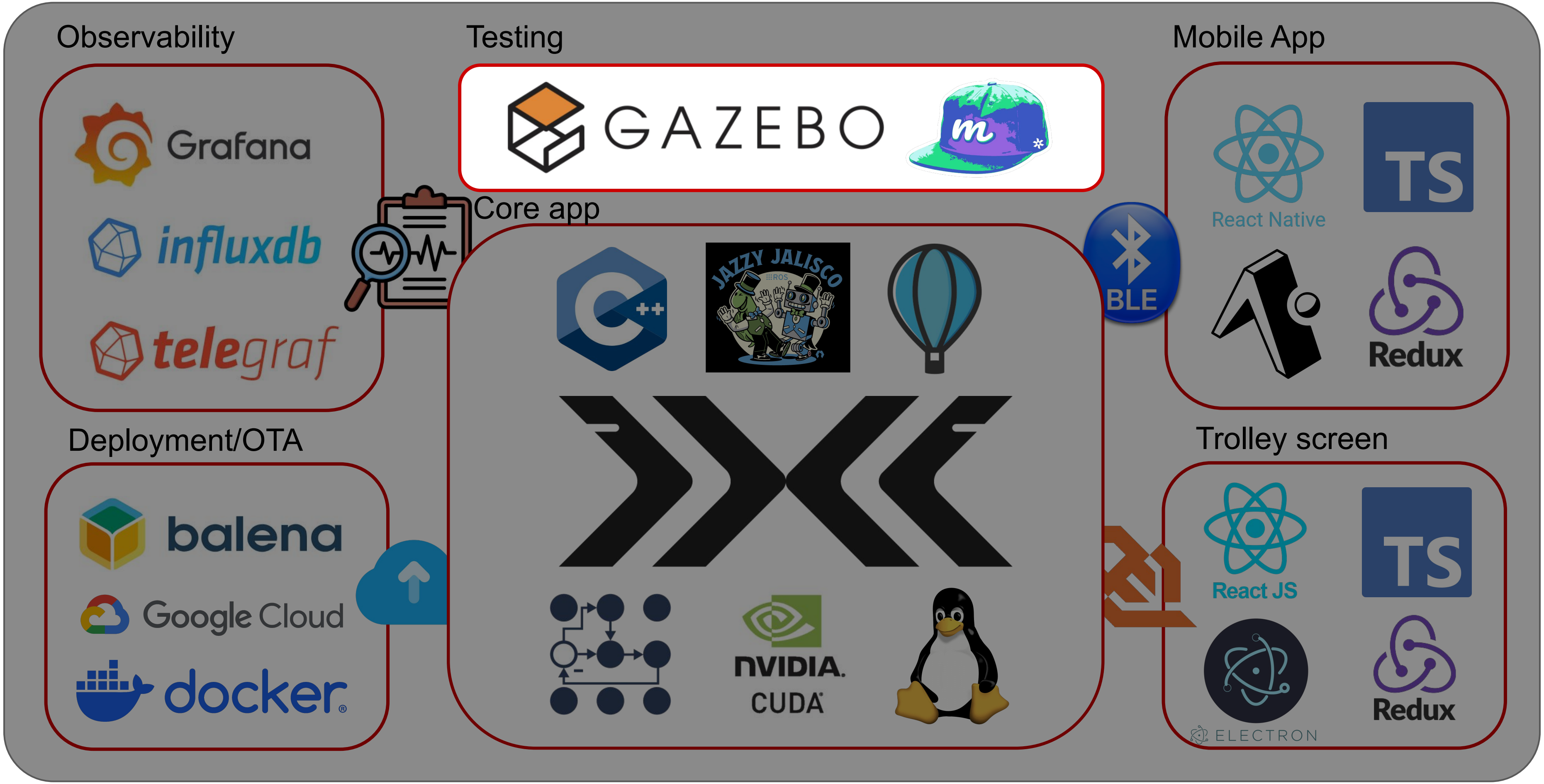
TEST



- Vscode devcontainer
- Ubuntu laptop
- NVIDIA GPU

- Vscode devcontainer
- SSH config
- OS on Jetpack

Balena dev trolley



Observability

Testing

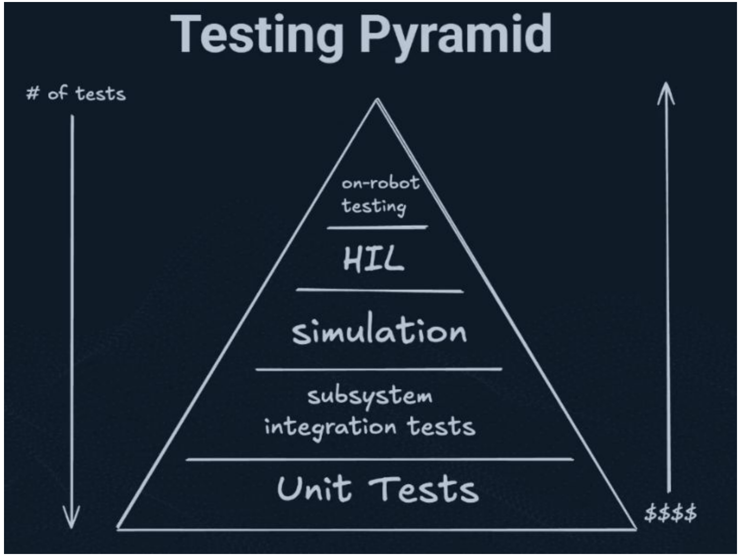
Mobile App

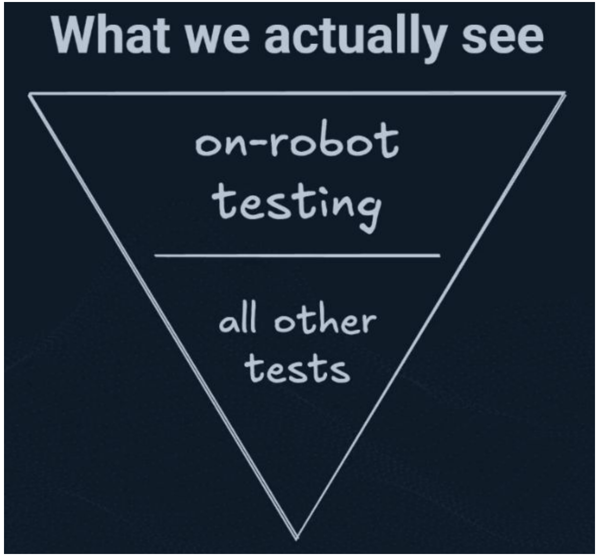
Core app

Deployment/OTA

Trolley screen

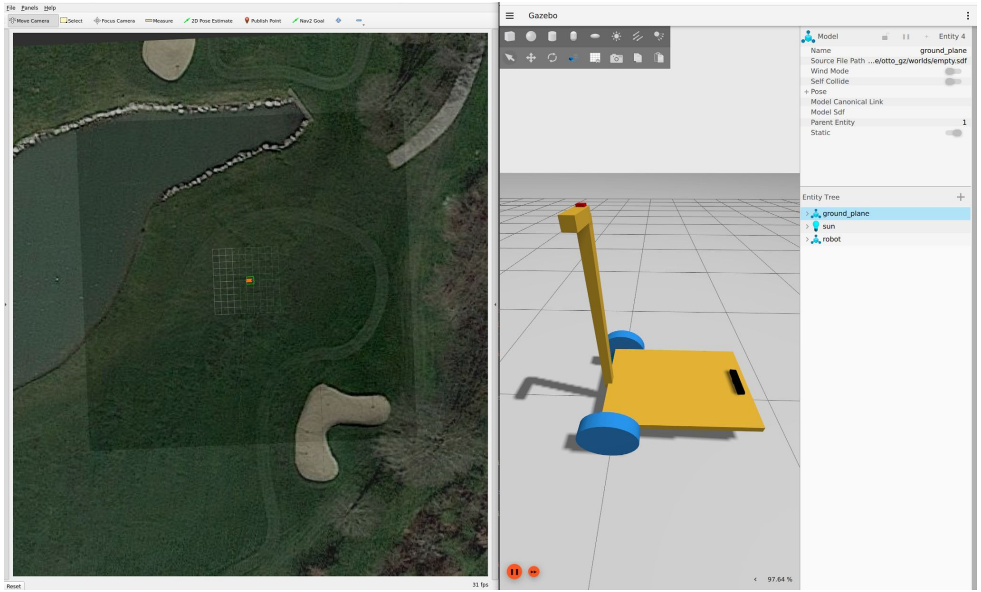
Testing objectives





Source : “Replay Testing Fast, Iterative Robotics Testing” @ ROSCON 2025
https://vimeo.com/1136204393

Where we are today: Simulation



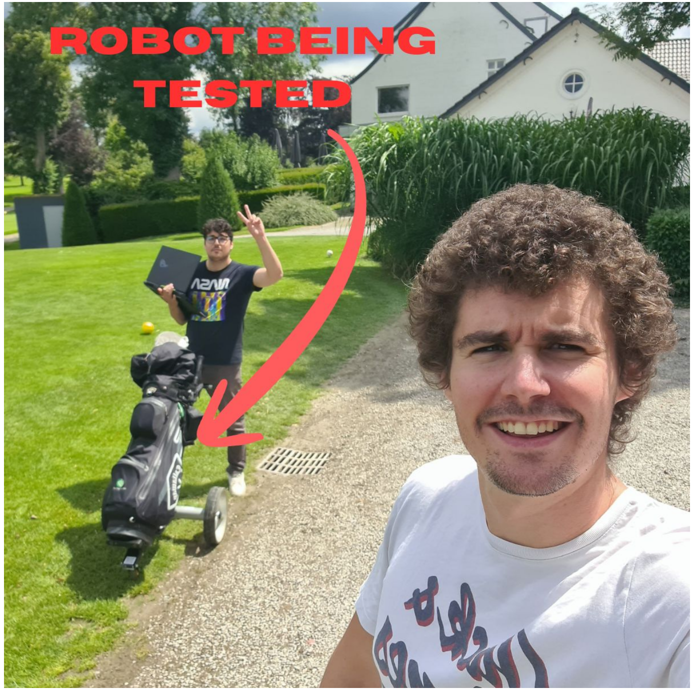
Name ground_plane
Source File Path ...e/otto_gz/worlds/empty.sdf
Wind Mode
Self Collide
+ Pose
Model Canonical Link
Model Sdf
Parent Entity 1
Static

- Can test localization
- Navigation like go to GPS

- Can’t test follow
- Can’t test vision
- Can’t test robot dynamics

→ Need to upgrade the simulation and test integration

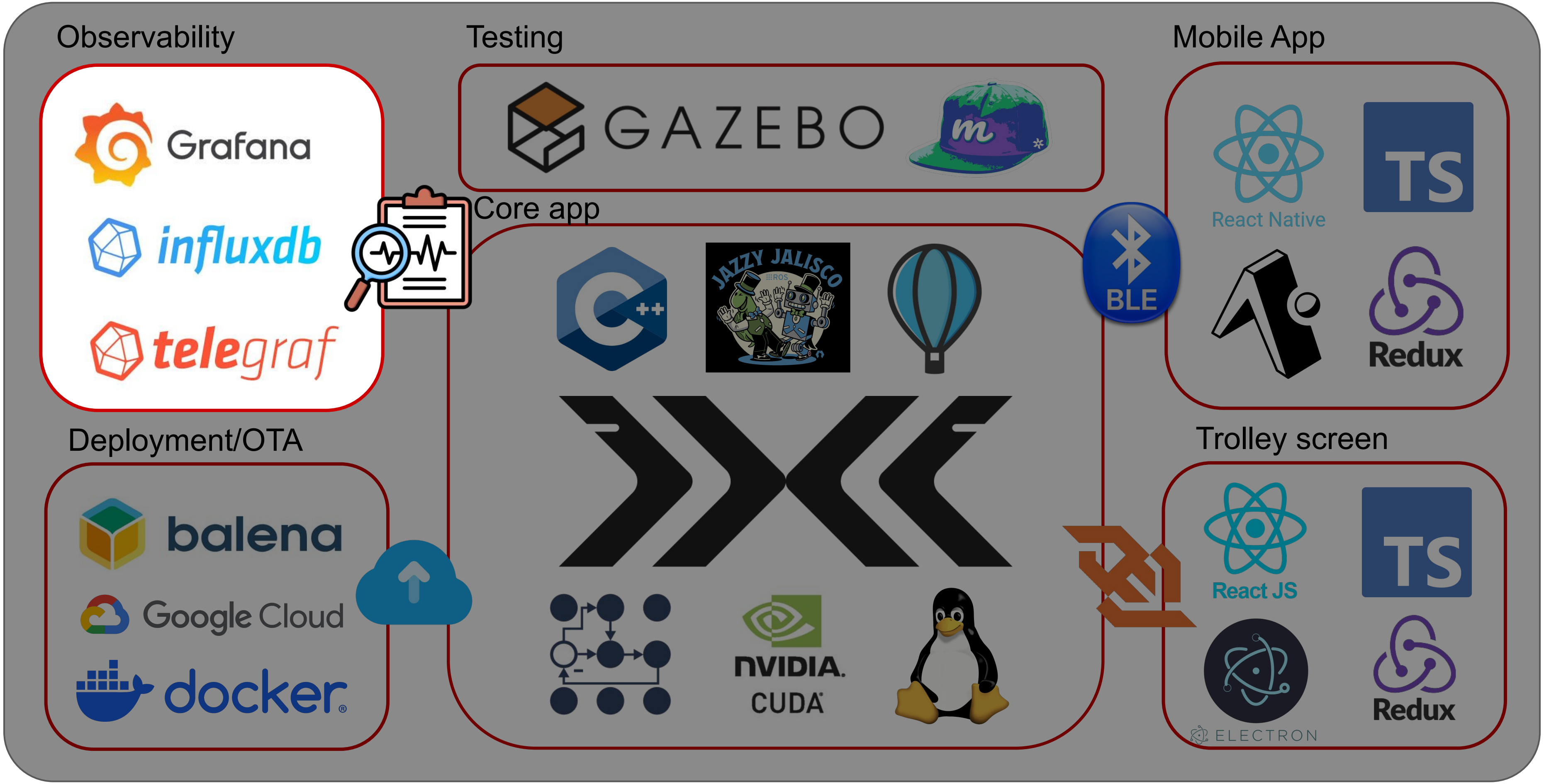
Where we are today: On Field



Live debugging: Rviz + Plotjuggler + RQT over VNC

Bags:

- Custom bag recorder (rolling windows) based on user inputs
- UI button to report bags (for end users)
- Download the bag remotely (no upload system yet) for support


Observability

Testing

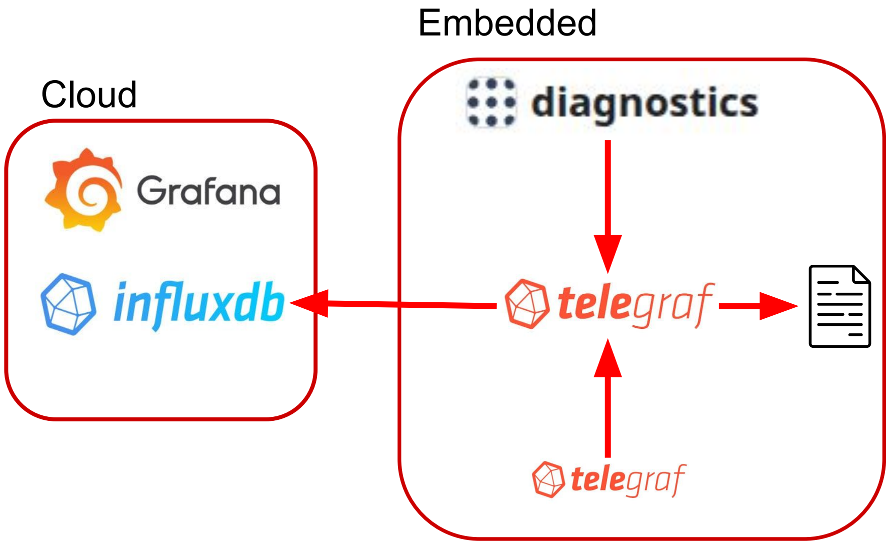
Mobile App

Core app

Deployment/OTA

Trolley screen

Observability - TIG stack



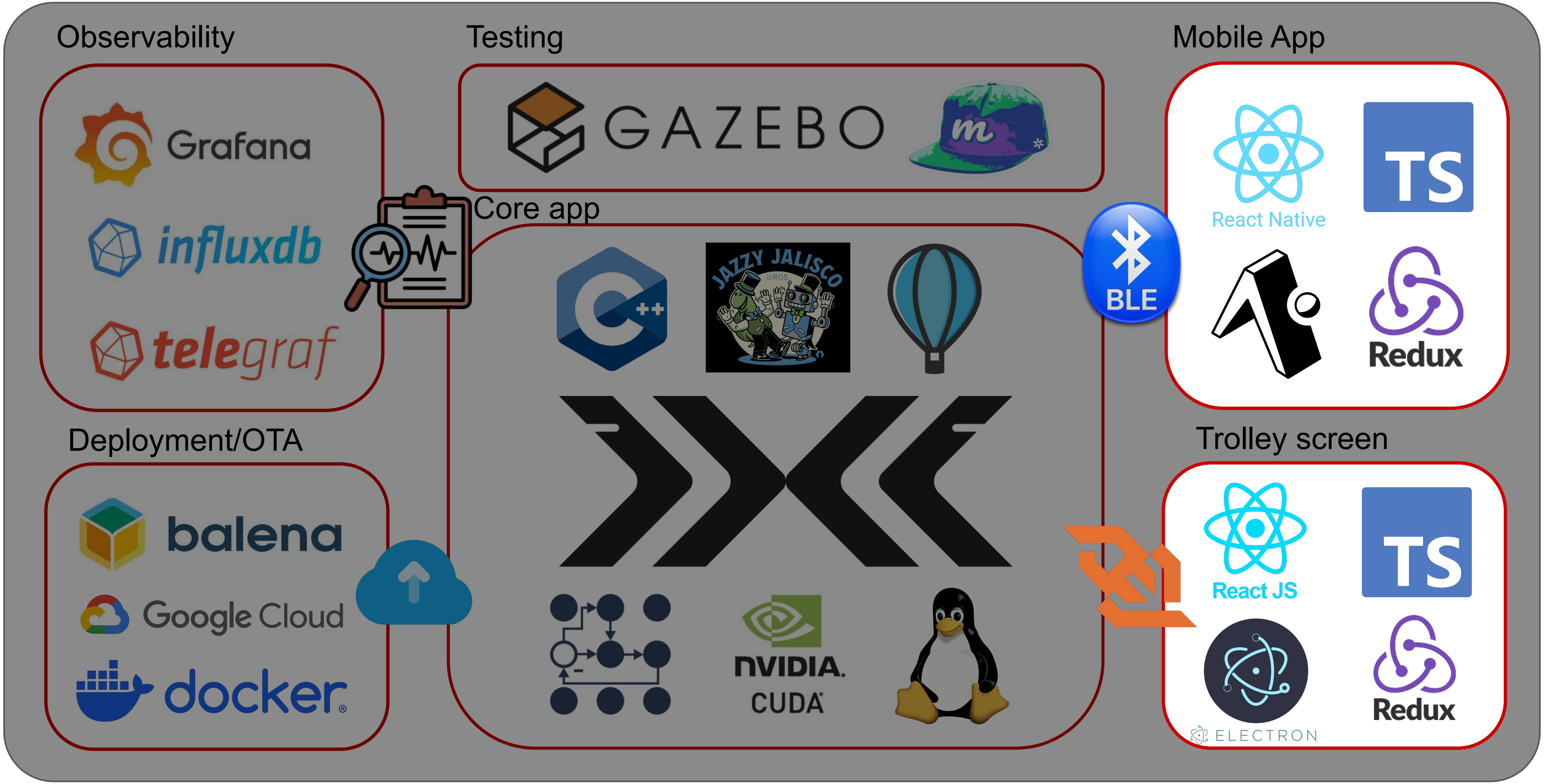
ROS Diagnostics:

- UI over VNC
- Common + Custom diagnostics
- diagnostic_remote_logging

File:

- When no 5G connection
- Analysis with LLMs
- Helped us exploring tool by tool

Pricing !!! We avoid $/device



Observability

Testing

Mobile App

Core app

Deployment/OTA

Trolley screen

User interfaces

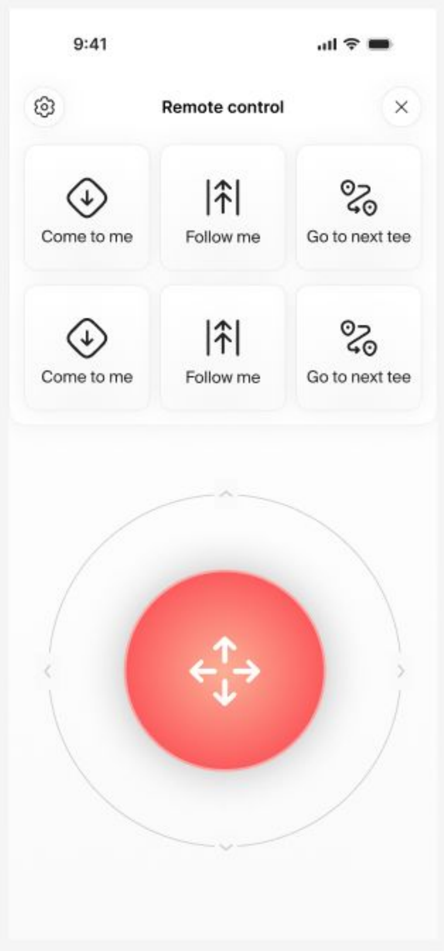
- We have two UI applications
  - Mobile Application (first one developed)
  - Tactile Screen Application
- General rule: no ROS in front
  - We need a bridge even for embedded screen
  - Keep standards development architecture
  - Separate concerns to not be too lock with specific solutions

Hope to avoid it in the future (communication through the middleware directly)

Mobile application

We have tried:

- Rosbridge server + wifi hotspot
  - not suitable for B2C
  - connect smartphone to hotspot is not user friendly
  - Too much ROS concept in the communication
- Bluetooth classics
  - Issue with IOS and not as maintained as BLE

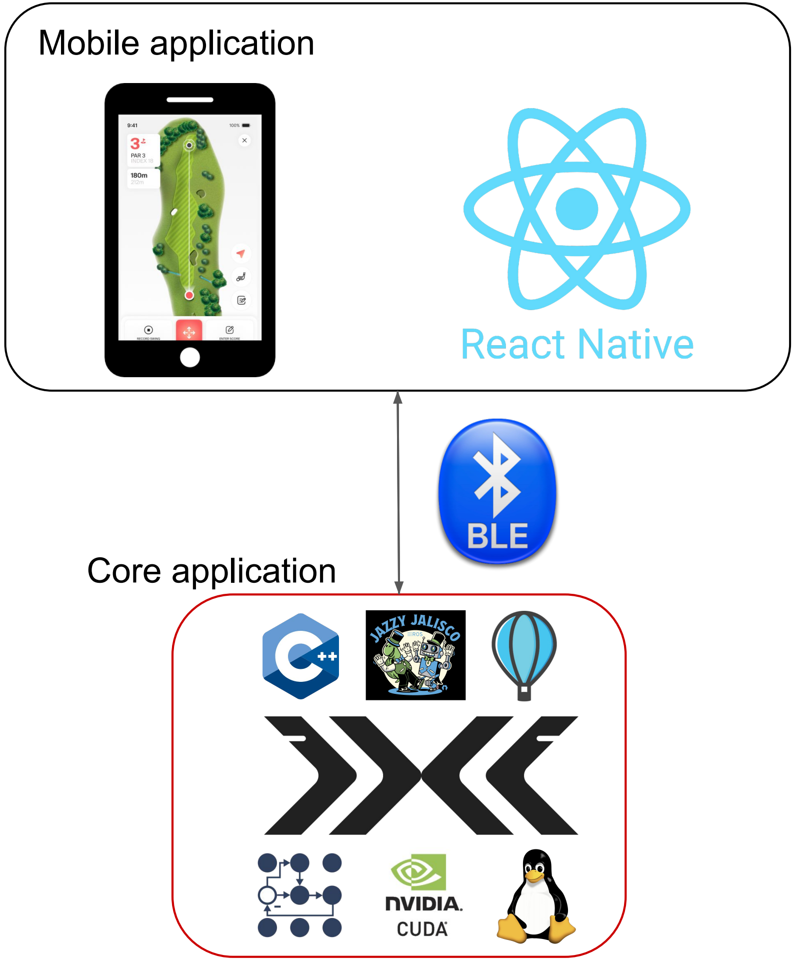


Mobile application

We chose Bluetooth Low Energy

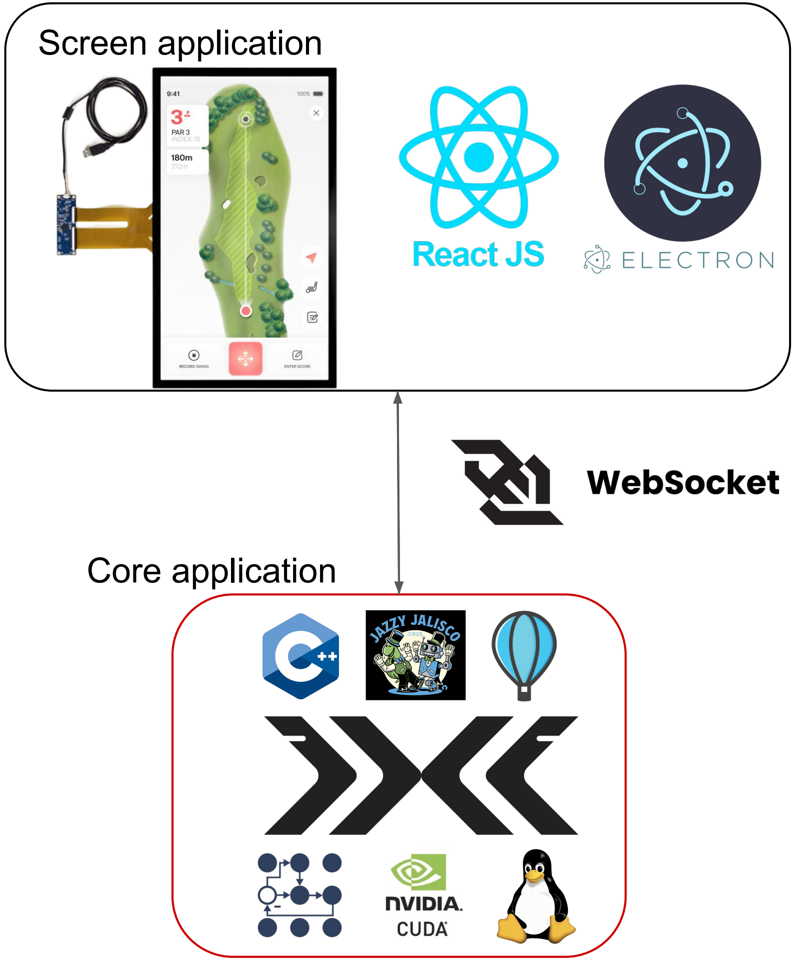
- 100m with recent specifications
- Lack of resources
- Currently using Bless
- Our first use case for rclrs with BlueR?

No maintained ros2 package to bluetooth (BLE bridge/api)



Tactile screen application

- Custom ws bridge to communicate with ROS 2 stack:
    - Separate concerns: bring robotics/ROS logic in the frontend
- The application runs in a Docker container locally
- Electron to develop a desktop app
    - Access to the file system
    - Easy to release as AppImage on Linux



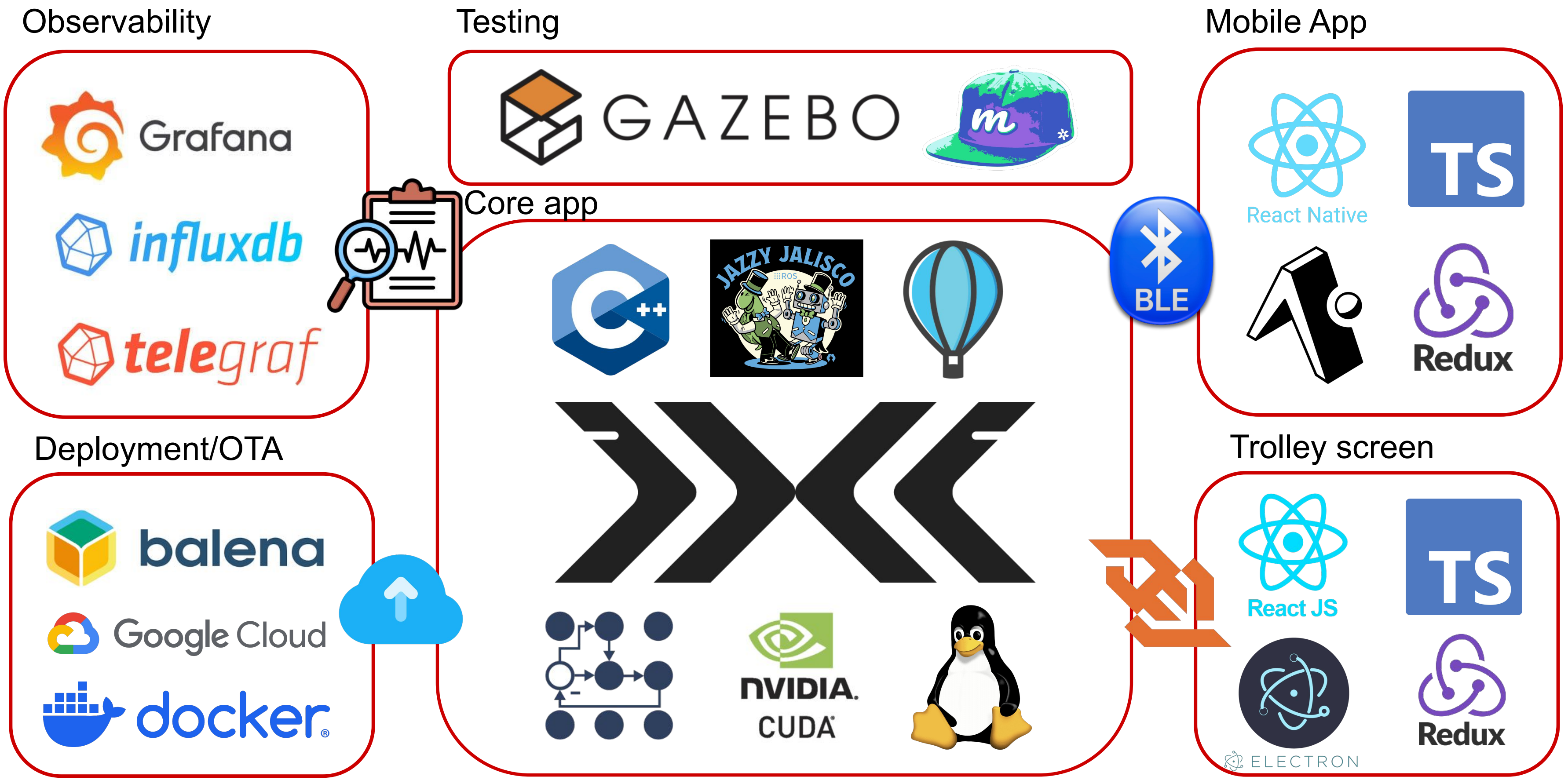
Screen application

React JS

ELECTRON

WebSocket

Core application


Observability

Testing

Mobile App

Core app

Deployment/OTA

Trolley screen

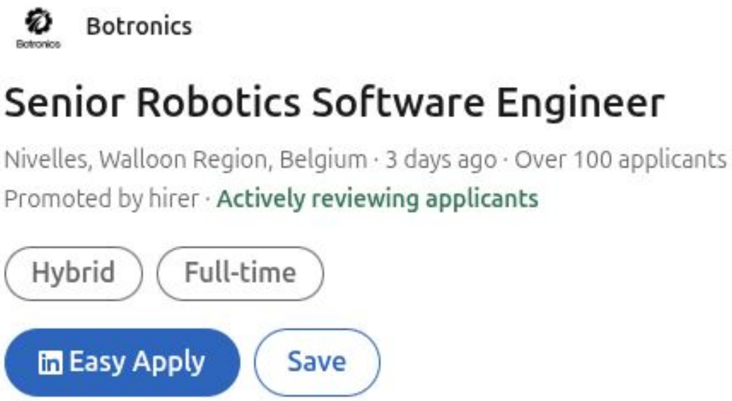
ROSCon Belgium is happening in 2026!

- In conjunction with the OSRF
- 25 & 26 november 2026
- Nivelles/Nijvel (40 min from here)
- In English
- Speakers & Sponsors wanted !

*

* Logo not yet confirmed

We are hiring !



Botronics

Senior Robotics Software Engineer

Nivelles, Walloon Region, Belgium · 3 days ago · Over 100 applicants

Promoted by hirer · Actively reviewing applicants

- ROS2/Nav2
- C++ (Rust bonus)
- Experience in production deploymen
- Startup mindset
- English (French bonus)
- Passion for robotics

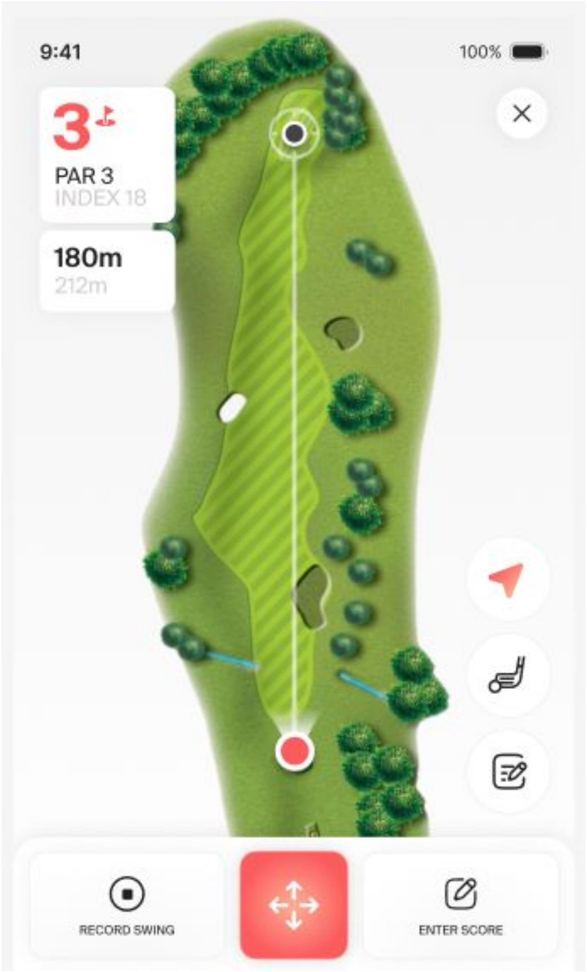
Q&A

Appendices

Tactile screen application

An application is running on the screen of the robot which allow the user to interact with the Robot

- Send goals on a custom map
- Trigger navigation behaviors
- Visualize data from the game
- Videos preview and recording during the game



Why ROS 2 ?

SOMETIMES IT'S TOO SLOW, FOR SURE
IT NEEDS TO BE REFACTORED, FOR SURE
imgflip.com