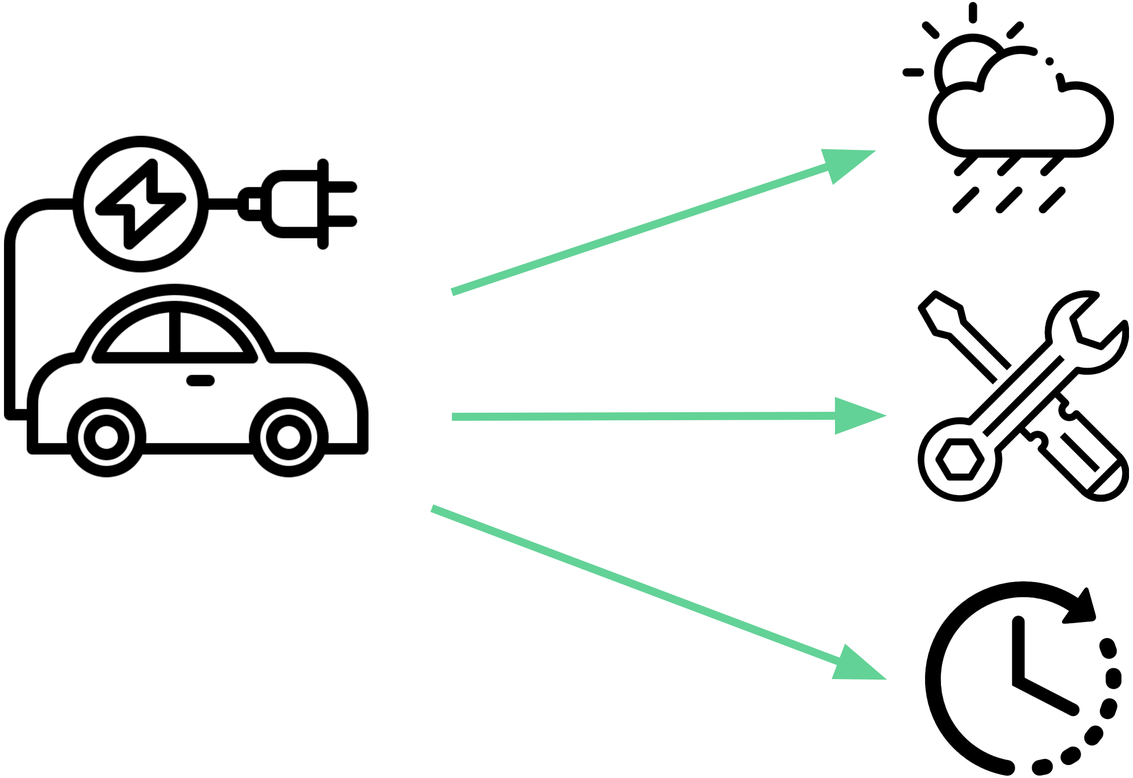
ME599 Project Presentation

Dockerization of CAV Codebase & CARLA

By: Urban Pistek

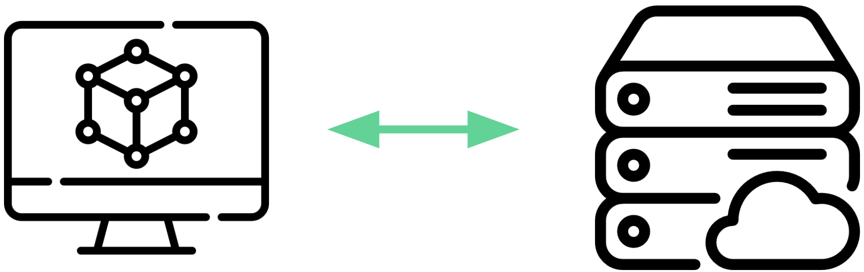
Background: Vehicle Software Development



Background: Vehicle Software Development

Need: Develop and test software without conflicting with rest of the team, in a controlled environment.

Solution: Leverage simulation and software tools to automate testing and parallelize development.



Background: Software & Simulation Tools

Solution: Leverage simulation and software tools to automate testing and parallelize development.

Engineering Methodology: Docker

Docker: A software platform that allows developers to easily create, deploy, and run applications in a virtual environment called a container.

Portability: Docker containers are platform-agnostic

Efficiency: Lightweight and use fewer resources than traditional virtual machines

Consistency: Built from a set of instructions called a Dockerfile, which ensures that the application is built and deployed in a consistent and repeatable way

Isolation: Docker containers provide a level of isolation between the application and the host system

Scalability: Docker makes it easy to scale applications horizontally by running multiple instances of the same container across multiple machines

Engineering Methodology: CARLA

CARLA: An open-source simulation platform designed for testing and developing autonomous driving systems.

Semi-Realistic Environment: Realistic and configurable simulation environment that includes realistic physics, weather conditions, and a range of urban and suburban scenarios

Safe & Cost-Effective Testing: Test autonomous driving algorithms and systems in a safe and controlled environment, without the need for real-world testing

Sensor Simulation: Supports lidar, radar, and cameras

Open-source & Extensible: Open-source platform, which means it can be customized and extended to meet the specific needs of individual developers and research teams

Engineering Methodology: ROS

ROS: ROS (Robot Operating System) is an open-source framework for building and programming robots

Modularity: Easy to reuse code and build on existing components

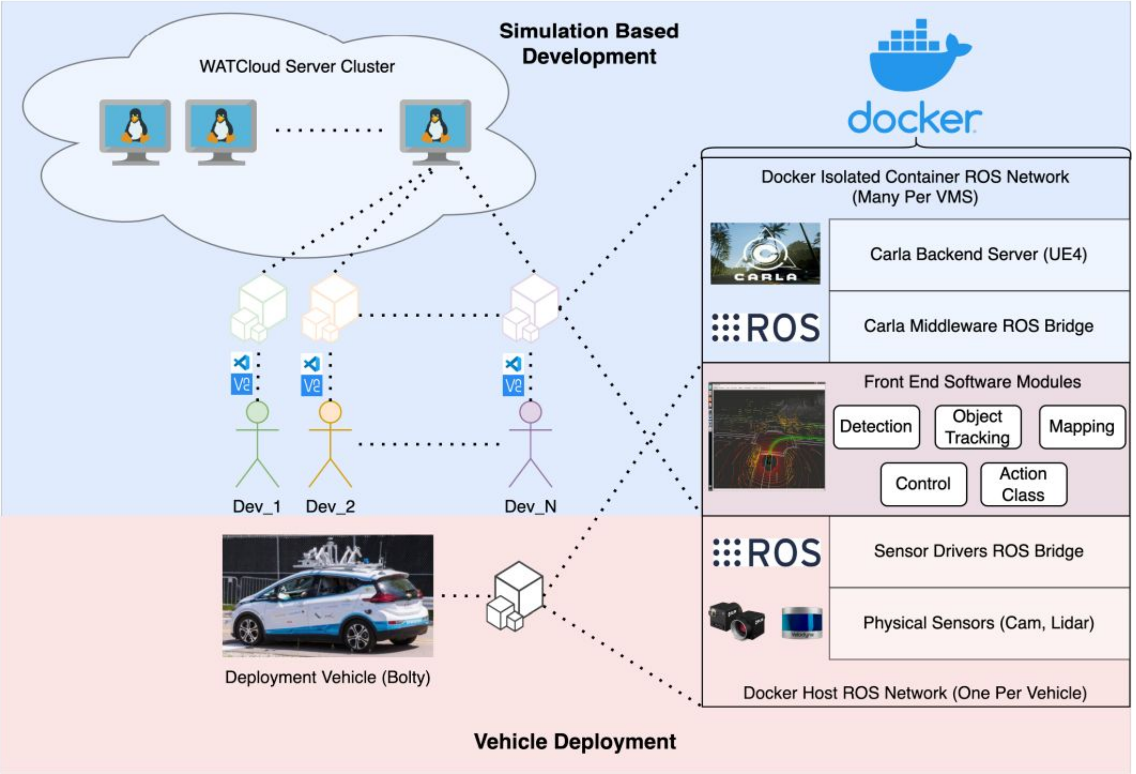
Flexibility: Supports a wide range of hardware and software platforms

Large Community: Active community of developers, researchers, and users

Rich set of tools and libraries: Visualization tools, simulation environments, and software libraries

Open-source & Extensible: Open-source, which means it can be customized and extended to meet the specific needs of individual developers and research teams

Engineering Methodology: Architecture

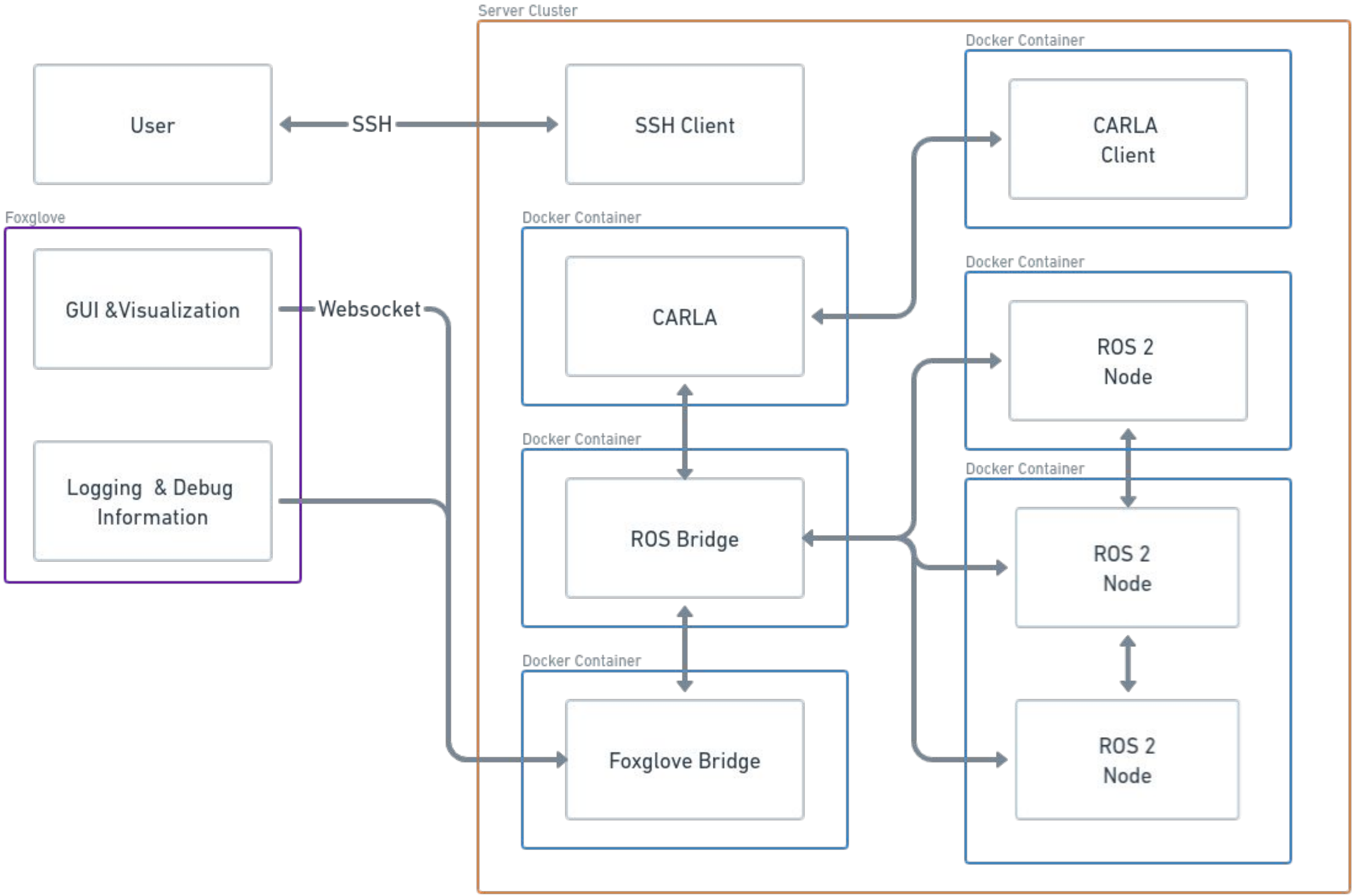


[1] WATonomous Software Architecture and Data Pipeline

Engineering Methodology: Architecture

Requirements:

1. Utilize ROS, CARLA and Docker and integrate them together
2. Able to run locally or in a server/cloud setting
3. Minimal environment setup / configuration - just deploy and run
4. Modular, able to run nodes separately
5. Able to scale by adding more nodes or tools
6. Can setting GUI / ROS data from server to local machine in real-time


Server Cluster
User
SSH
SSH Client
Docker Container
CARLA Client
Foxglove
Docker Container
GUI &Visualization
Websocket
CARLA
Docker Container
ROS 2 Node
Docker Container
Logging & Debug Information
ROS Bridge
Docker Container
ROS 2 Node
Docker Container
Foxglove Bridge
ROS 2 Node

Challenges: Development

1. Building the ROS-Carla Bridge for ROS 2
2. Getting the ROS-Carla Bridge to connect to the Carla server
3. Compatibility between ROS / Carla / Python versions and releases
4. Getting a feed of the GUI remotely from the server instance
5. Using Foxglove for websocket data streaming

Challenges: Additional

1. Coordination with WATO for server access & other resources
2. Learning and getting familiar with CARLA
3. Less documentation for ROS 2 & newer Carla releases

Integration: Tools & Software Used

foxglove

CARLA

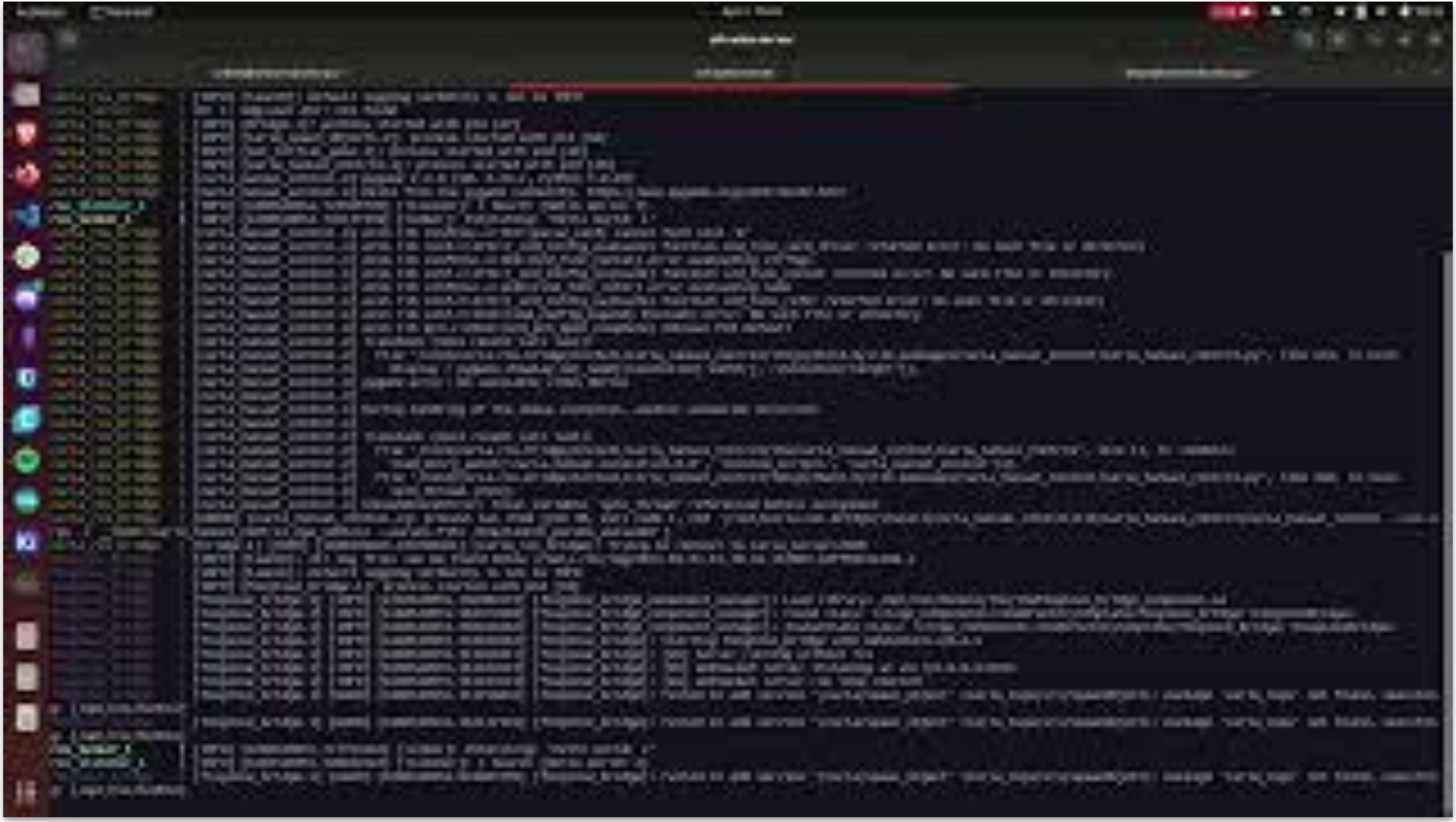
ROS

Integration: Demos

Demo 1

Demo 2

Demo 3



Recommendations: Improvements

1. Better configuration of Foxglove studio
2. Streamline docker images (optimize, make smaller)
3. Debug initial Carla-ROS-Bridge connection issue (Issue#673)
4. Make wrapper to make deployment more configurable (Similar to watod)
5. Work and collaborate with WATO more (Get server access, Join discord, join syncs with their simulation team members)

Lessons Learned: Knowledge

1. Better understanding of Carla, Docker & ROS
2. Better understanding of deploying docker clusters
3. Better understanding of dockerizing software & tools
4. Connect with experts and knowledgeable people early on
5. Collaborate everywhere possible

Summary: Links

[Find Example Project on Gitlab]

[Access to WATO Server]

[Running Carla in Docker]

[Foxglove Websocket Bridge]

Questions?