ROS 2 Action Server and Client Report

Student: Temur Akhrorov

Student ID: 12204574

\*\*Table of Contents\*\*

1. Introduction

2. Prerequisites

3. Installation

4. Usage

a. Running the Action Server (C++)

b. Running the Action Client (C++)

5. Action Definition

6. Customization

7. License

8. Acknowledgments

--------------------------------------------------

\*\*1. Introduction\*\*

This report discusses the setup of a ROS 2 action server and client for simulating an order delivery system. We use the "OrderDelivery" action defined in the `delivery\_action\_interfaces` package to perform deliveries. This system can be customized for real-world delivery scenarios.

\*\*2. Prerequisites\*\*

Before you begin, make sure you have the following prerequisites:

- ROS 2 installed and set up.

- A ROS 2 workspace with the `delivery\_action\_interfaces` package containing the "OrderDelivery" action definition.

\*\*3. Installation\*\*

Clone the repository into your ROS 2 workspace:

```bash

cd <your\_ros2\_workspace>/src

git clone <repository\_url>

Build the workspace:

cd <your\_ros2\_workspace>

colcon build

**4. Usage**

**a. Running the Action Server (C++)**

To run the C++ action server for order delivery:

bashCopy code

ros2 run <your\_package\_name> order\_delivery\_server

The server is now ready to accept delivery orders.

**b. Running the Action Client (C++)**

To run the C++ action client and send a delivery request:

bashCopy code

ros2 run <your\_package\_name> order\_delivery\_client

The client will send a delivery request to the action server and display the result.

**5. Action Definition**

The "OrderDelivery" action definition is located in the **delivery\_action\_interfaces** package. It comprises three parts: Request, Result, and Feedback.

* Request: Specifies the order to be delivered and the destination.
* Result: Indicates whether the delivery was successful and provides a message.
* Feedback: Includes the progress of the delivery.

**6. Customization**

You can customize the action server logic in the **order\_delivery\_server.cpp** file and adapt the system for specific delivery scenarios.

**7. License**

This project is licensed under the MIT License. For details, refer to the [LICENSE](https://chat.openai.com/c/LICENSE) file.

**8. Acknowledgments**

Special thanks to the ROS 2 community for providing a robust platform for developing robotic applications.

This report provides an overview of the ROS 2 action server and client setup for order delivery. It covers the necessary steps for installation, usage, customization, and acknowledges the contributions of the ROS 2 community. This setup can serve as a foundation for building real-world robotic delivery systems using ROS 2.

csharpCopy code

The Python part has been removed from the report.