Lab02_PubSub

Pia Dünow (s0582141)

Scenario

I implemented a basic Pub/Sub message/ticker system where publishers can send messages to various channels (e.g. sport, news), and subscribers can subscribe to these channels to receive messages relevant to them. The scenario includes:

1. **Message Broker**: Acts as the central hub for handling subscriptions and message distribution.
2. **Publishers**: Send messages to specific channels.
3. **Subscribers**: Subscribe to channels and receive messages published on those channels.

Programming Language

The implementation is done in Python, utilizing its `asyncio` library to handle asynchronous I/O operations

Theoretical Knowledge

The Pub/Sub model is a key concept in distributed systems, enabling efficient and scalable communication. This model's decoupling of publishers and subscribers enhances flexibility and scalability, allowing each component to operate independently. My implementation demonstrates fundamental concepts such as:

1. **Decoupling**: Publishers and subscribers do not need to know about each other's existence.
2. **Asynchronous Communication**: Both publishers and subscribers operate asynchronously, ensuring non-blocking communication.
3. **Event-Driven Architecture**: The broker acts as an intermediary, facilitating an event-driven communication model where subscribers react to events (messages) published by publishers.

Implementation Details

The implementation consists of three main scripts:

1. **Broker Script (`broker.py`)**: This script sets up a server to handle both publishers and subscribers. It maintains a dictionary to track channels and their respective subscribers. When a message is published, it distributes the message to all subscribers of the specified channel.
2. **Publisher Script (`pub.py`)**: This script allows users to publish messages to specific channels. Users can input the channel name and message, which are then sent to the broker.
3. **Subscriber Script (`sub.py`)**: This script enables users to subscribe to multiple channels. Users can specify channels one by one, and the script will listen for messages from these channels.

ressources used: ChatGPT