Addis Ababa Institute of Technology

School of Information Technology and Engineering

Department of Software Engineering (Information Technology Stream)

# Fundamentals of Distributed Systems

Lab 4: Answers to reflection questions

Prepared by: Nathnael Getachew

ID: UGR/8932/13

Submission Date: 13/11/2024

## Answers to reflection questions

1. **How does the buffer size affect the frequency and timing of message passing?**
   **Answer:**
   - The Buffer Size effect on frequency:
     - Unbuffered Channels: Unbuffered channels in Go necessitate simultaneous readiness on the part of the transmitter (producer) and receiver (consumer) in order for a message to be transmitted. This implies that until the customer is prepared to receive the message, the producer will block. Consequently, the consumer's capacity to process messages is closely linked to the frequency of message passing. The producer will be blocked if the consumer is slow, which could cause delays in the generation of messages.
     - Buffered Channels: These channels don't require an immediate recipient and let a certain quantity of messages be sent. The number of messages that can be held in the channel before the producer is blocked depends on the buffer size. A bigger buffer size enables the producer to send more messages before it has to wait for the consumer to catch up.
   - The buffer size effect on Timing:
     - With Small Buffers: If the buffer size is small, the producer may frequently block, waiting for the consumer to read messages.
     - With Large Buffers: A larger buffer allows the producer to send messages at a faster rate than the consumer can process them, leading to a more asynchronous operation. The producer can continue to send messages until the buffer is full, which can improve the overall efficiency of the system.
2. **What Happens When the Buffer is Full?**
   **Answer:**
   When the buffer is filled, the producer's behaviour changes:
   - The producer will halt (that is, stop executing) until there is enough space in the buffer. This means that if the consumer does not consume messages quickly enough to clear space in the buffer, the producer will be unable to send fresh messages. This can result in a situation where the producer is essentially waiting for the consumer to catch up, which can cause delays in the entire message-passing process.
3. **How does RabbitMQ handle load balancing between multiple consumers?**
   **Answer:**

   - RabbitMQ distributes messages to consumers in a round-robin fashion. When multiple consumers are connected to the same queue, RabbitMQ will send the next message to the next available consumer, effectively balancing the load.

4. **What happens when a consumer disconnects?**
   **Answer:**
   - If a consumer disconnects, RabbitMQ will stop sending messages to that consumer. Any unacknowledged messages will remain in the queue and can be delivered to other consumers. If the consumer was set to acknowledge messages manually, those messages will not be removed from the queue until acknowledged.

5. **How does NATS handle different subjects?**
   **Answer:**
   - NATS uses a publish-subscribe messaging model where messages are sent to subjects rather than specific queues or consumers. Each subject acts as a distinct channel for messages, allowing publishers to categorize messages based on their context or purpose.
6. **What advantages does this give in message organization?**
   **Answer:**
   The ability to use different subjects in NATS provides several advantages for message organization:

   - Categorization of Messages: Allows us to categorize the messages logically. This makes it easier for developers to manage and understand the flow of messages, as related messages can be grouped under common subjects.
   - Selective Message Delivery: Allows subscribers to select the messages they would like to receive. This reduces unnecessary data processing and network traffic, as subscribers are not overwhelmed with irrelevant messages.
   - Scalability: As new subjects are added without altering the existing infrastructure. This means that new features or components can be added without any problems.
   - Dynamic Subscription: With the ability to create dynamic subscriptions based on subjects, applications can adapt to changing requirements. For example, if a subscriber needs to receive error messages, it can simply subscribe to the updates.error subject without needing to modify the publisher or other subscribers.