

# Асинхронна передача даних в мікросервісних архітектурах за допомогою черг повідомлень

Стасько Тарас

Науковий керівник: Глибовець А.М.  
Доктор технічних наук, Декан  
Факультету Інформатики

# Актуальність теми

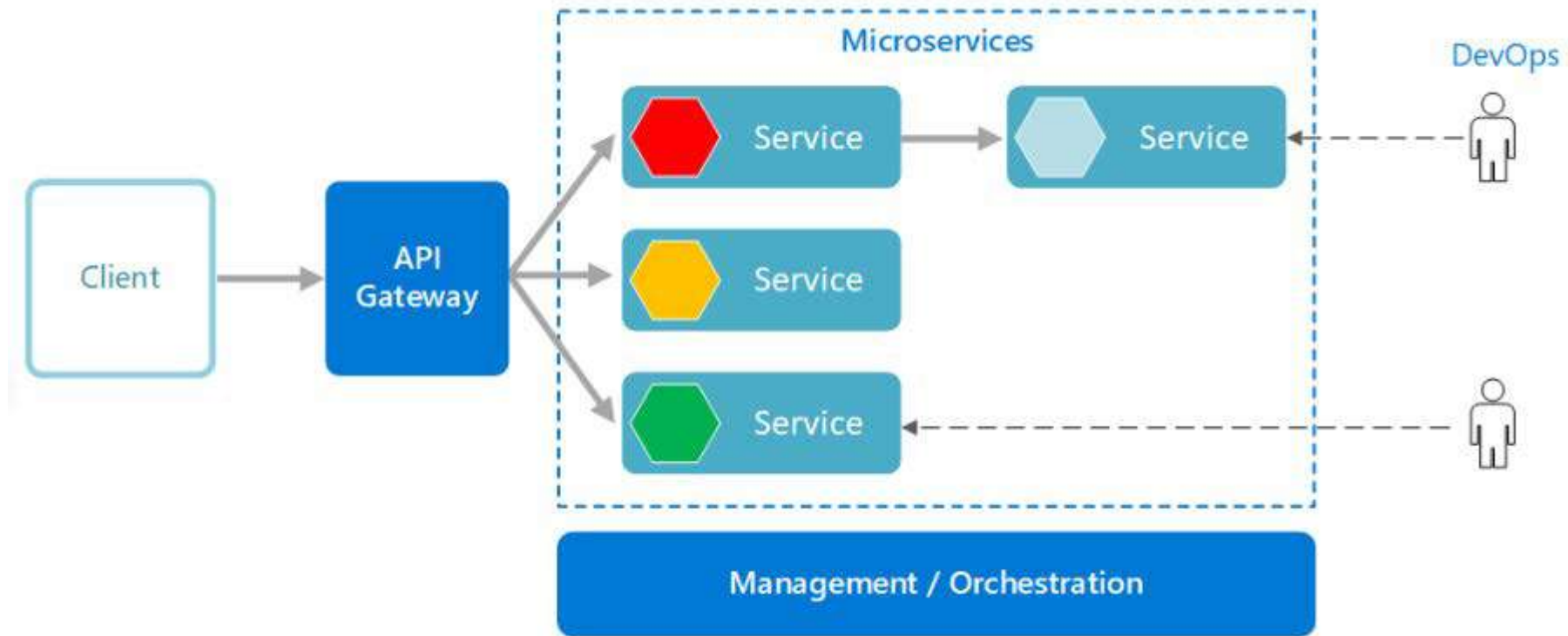
**Синхронна комунікація** в мікросервісах призводить до тісного зв'язку між сервісами, каскадних збоїв, складностей масштабування.

**Асинхронна передача даних** через черги повідомлень допомагає вирішити ці проблеми

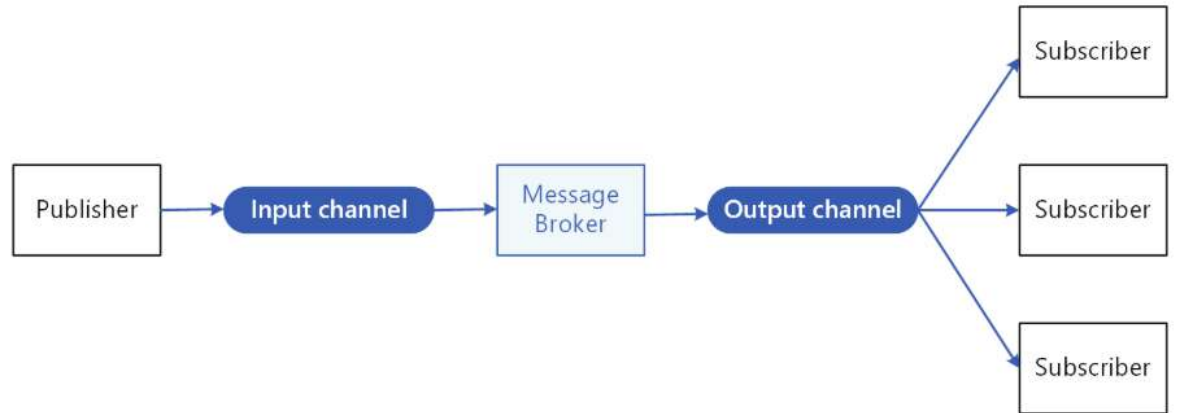
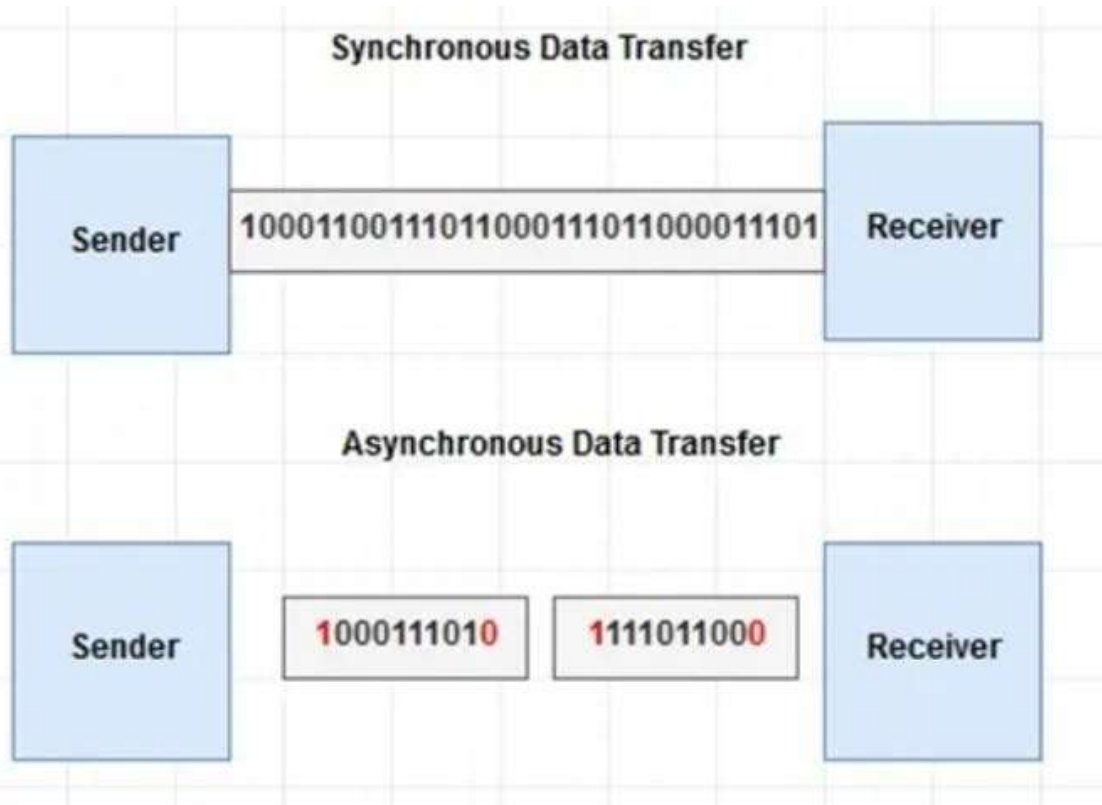
# Мета та завдання дослідження

- Дослідити принципи асинхронної передачі даних у мікросервісах та визначити оптимальні підходи.
- Завдання:
  - Аналіз проблем синхронної комунікації.
  - Дослідження асинхронних патернів (Pub/Sub, Saga, CQRS).
  - Порівняння брокерів повідомлень.
  - Оптимізація продуктивності та безпеки.

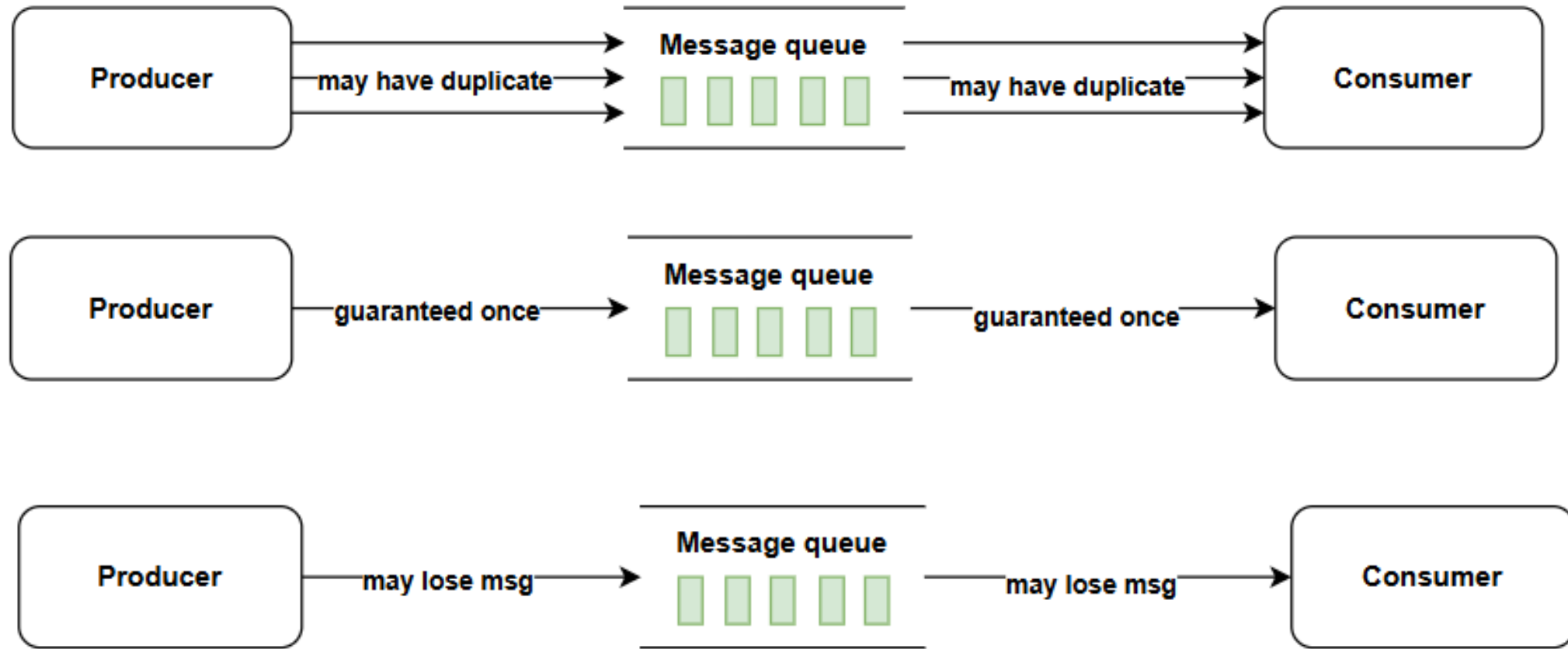
# Мікросервісна архітектура



# Асинхронна передача даних



# Черги повідомлень



# Практична реалізація

## Мета:

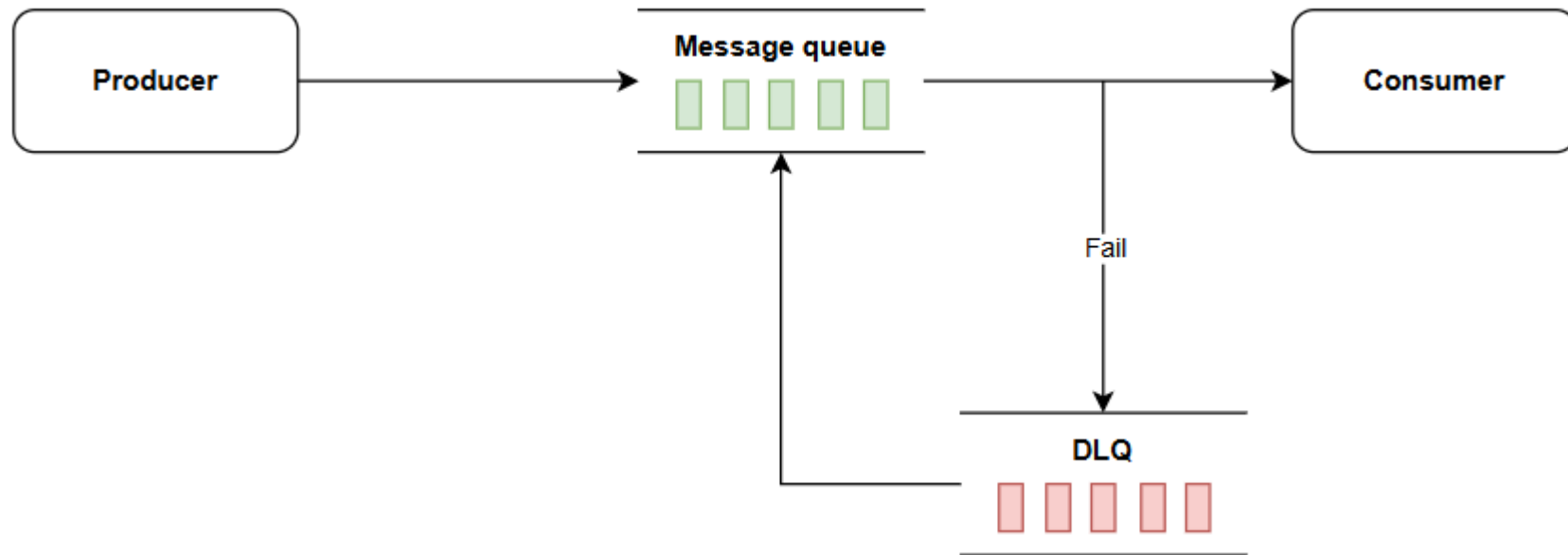
Реалізувати обробку повідомлень через чергу з підтримкою повторних спроб і черги мертвих листів (DLQ).

## Компоненти:


- **FastAPI Producer:** генерує та надсилає замовлення в RabbitMQ.
- **RabbitMQ Broker:** обробляє черги, маршрутизацію та повторні спроби.
- **Async Consumer:** читає повідомлення з черги, обробляє з можливими помилками.
- **Dead-Letter Queue (DLQ):** отримує повідомлення, які не вдалося обробити після кількох спроб.

# Практична реалізація

Схема взаємодії:



# Робота системи

 RabbitMQ™ RabbitMQ 4.1.0 Erlang 26.2.5.11

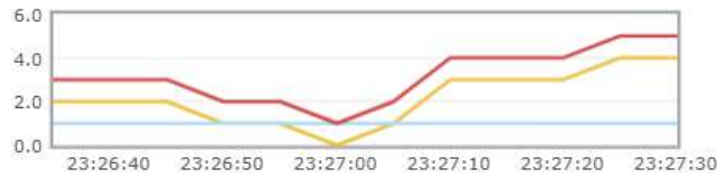
⚠ All stable feature flags must be enabled after completing an upgrade. [\[Learn more\]](#)

[Overview](#) [Connections](#) [Channels](#) [Exchanges](#) **[Queues and Streams](#)** [Admin](#)

## Queue order\_created

▼ Overview

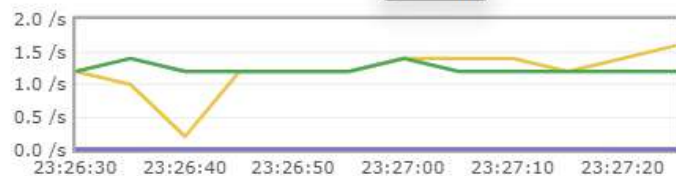
Queued messages [last minute](#) ?



Ready 6  
Unacked 1  
Total 7

Message rates [last minute](#) ?

[No Title]




Publish 1.6/s  
Deliver (manual ack) 1.2/s  
Deliver (auto ack) 0.00/s

Consumer ack 1.2/s  
Redelivered 0.00/s  
Get (manual ack) 0.00/s

Get (auto ack) 0.00/s  
Get (empty) 0.00/s

Details

# Робота системи

 RabbitMQ™ RabbitMQ 4.1.0 Erlang 26.2.5.11

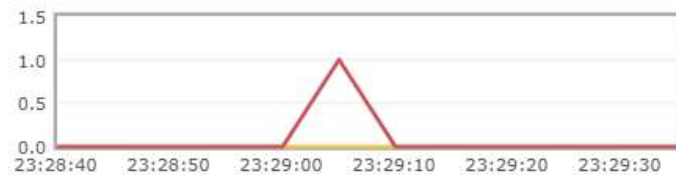
⚠ All stable feature flags must be enabled after completing an upgrade. [\[Learn more\]](#)

[Overview](#) [Connections](#) [Channels](#) [Exchanges](#) **[Queues and Streams](#)** [Admin](#)

## Queue order\_created\_dlq

▼ Overview

Queued messages [last minute](#) ?



Ready 0  
Unacked 0  
Total 0

Message rates [last minute](#) ?

[No Title]



Publish 0.00/s  
Deliver (manual ack) 0.00/s  
Deliver (auto ack) 0.00/s

Consumer ack 0.00/s  
Redelivered 0.00/s  
Get (manual ack) 0.00/s

Get (auto ack) 0.00/s  
Get (empty) 0.00/s

# Робота системи

```
Processing message: {'item': 'laptop', 'quantity': 2} (retries: 3)
❌ Error: Simulated processing error
🚨 Max retries exceeded. Sending to DLQ: {'item': 'laptop', 'quantity': 2}
🔄 Received message from DLQ at 2025-05-08T23:30:15.987786
🔄 Retrying message (attempt 0)
🔄 Message requeued to order_created
Processing message: {'item': 'tablet', 'quantity': 1} (retries: 0)
✅ Successfully processed: {'item': 'tablet', 'quantity': 1}
Processing message: {'item': 'laptop', 'quantity': 5} (retries: 0)
✅ Successfully processed: {'item': 'laptop', 'quantity': 5}
Processing message: {'item': 'tablet', 'quantity': 1} (retries: 0)
✅ Successfully processed: {'item': 'tablet', 'quantity': 1}
Processing message: {'item': 'phone', 'quantity': 3} (retries: 0)
✅ Successfully processed: {'item': 'phone', 'quantity': 3}
Processing message: {'item': 'laptop', 'quantity': 5} (retries: 0)
✅ Successfully processed: {'item': 'laptop', 'quantity': 5}
```

# Приклади коду

- Генерація замовлень

```
@app.on_event("startup")
async def startup_event():
    asyncio.create_task(create_and_send_orders())
```

- Відправка в чергу

```
await channel.default_exchange.publish(
    aio_pika.Message(
        body=message_body,
        delivery_mode=aio_pika.DeliveryMode.PERSISTENT
    ),
    routing_key="order_created"
)
```

# Приклади коду

- Логіка Retry:

```
if retries > MAX_RETRIES:  
    await send_to_dlq(message.body)  
else:  
    await retry_message(message.body, retries)
```

- Налаштування DLX у RabbitMQ:

```
arguments = {  
    "x-dead-letter-exchange": "dlx",  
    "x-dead-letter-routing-key": "dlq"  
}
```

# Використані джерела

- <https://learn.microsoft.com/en-us/azure/architecture/guide/architecture-styles/microservices>
- <https://peditaa.com/what-is-the-difference-between-synchronous-and-asynchronous-data-transfer/>
- <https://learn.microsoft.com/en-us/azure/architecture/patterns/publisher-subscriber>
- <https://blog.bytebytego.com/p/at-most-once-at-least-once-exactly>

Дякую за увагу!