
AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

# Java Message Service

http://galaxy.agh.edu.pl/~fmal/jms

**Filip Malawski**

**fmal@galaxy.agh.edu.pl**

# Wiadomości

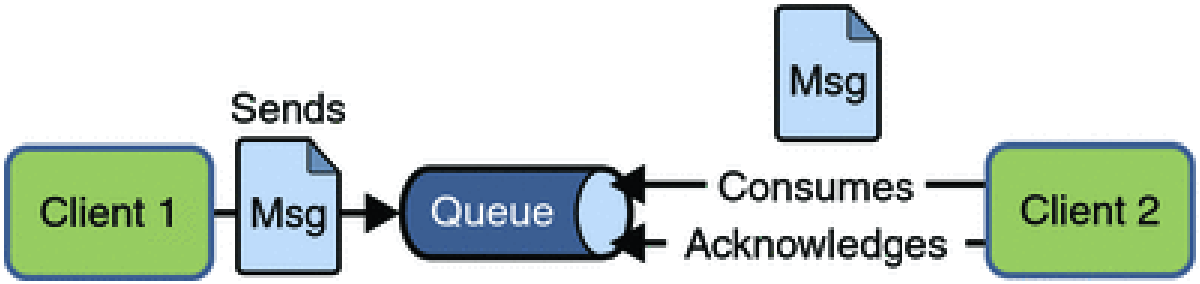
- Alternatywa dla wywołań metod
  - Format wiadomości zamiast interfejsu
  - Ukierunkowane na zdarzenia
  - Brak sztywnych zależności czasowych
- Luźne powiązania
- Komunikacja lokalna lub zdalna
- Message Oriented Middleware
  - Warstwa pośrednia dostarczająca mechanizmów obsługi wiadomości

# Modele komunikacji

- Komunikacja synchroniczna
  - Obie strony uczestniczące muszą być aktywne
  - Strona wysyłająca wiadomość otrzymuje potwierdzenie od odbiorcy
  - Wywołania blokujące
  - Np. chat
- Komunikacja asynchroniczna
  - Obie strony uczestniczące nie muszą być aktywne jednocześnie
  - Strona wysyłająca nie musi otrzymywać potwierdzenia odbioru
  - Wywołania nieblokujące
  - Np. e-mail

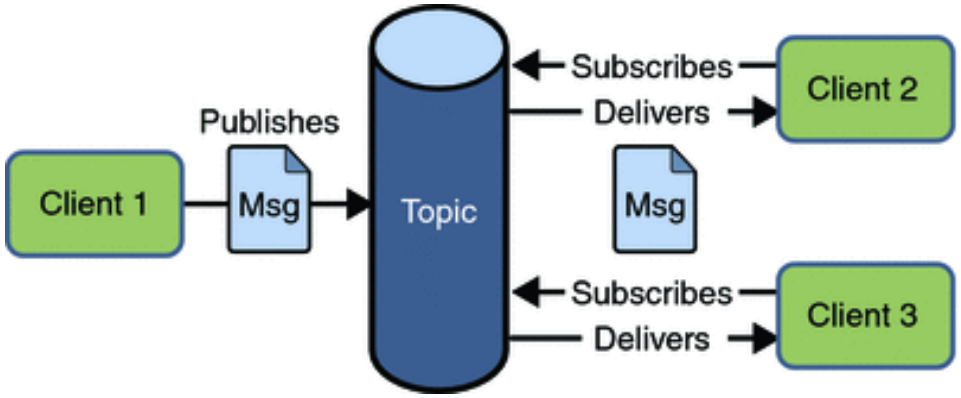
# Model Point-to-Point (PTP)

- Każda wiadomość ma tylko jednego konsumenta
- Brak zależności czasowych między nadawcą, a odbiorcą
- Konsument może pobrać wiadomość niezależnie od tego, czy był uruchomiony, gdy producent wysyłał daną wiadomość
- Odbiorca potwierdza udane przetworzenie wiadomości



# Model Publish-Subscribe (Pub\Sub)

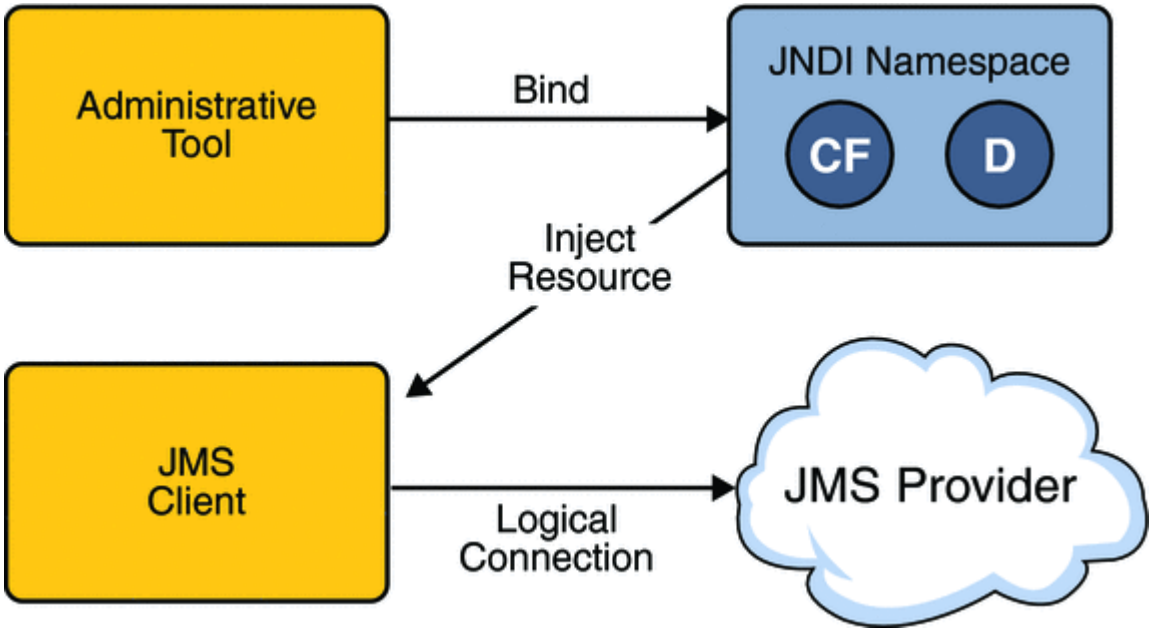
- Każda wiadomość może mieć wielu konsumentów
- Zależność czasowa pomiędzy nadawcami i odbiorcami
- Klient, który subskrybował się na dany temat może konsumować tylko wiadomości opublikowane po jego subskrypcji i tylko jeśli pozostaje aktywny



# Java Message Service

- Specyfikacja architektury umożliwiającej tworzenie, wysyłanie, odbieranie wiadomości.
- Wiele implementacji
- Nacisk na niezawodność oraz asynchroniczność
- Scentralizowane zarządzanie rozsyłaniem wiadomości

# Architektura



# Architektura

- **JMS provider** - system wiadomości implementujący interfejs JMS oraz dostarczający interfejsów administracyjnych
- **JMS clients** - programy lub komponenty napisane w języku
- Java, które produkują i konsumują wiadomości
- **Messages** - obiekty w których przesyłana jest informacja pomiędzy klientami JMS
- **Administered objects** - prekonfigurowane obiekty JMS stworzone przez administratora dla użytku klientów, rejestrowane w usłudze JNDI

  Dwa rodzaje: ***Destination*** oraz ***ConnectionFactory***
- **Native clients** - programy, które używają natywnego API produktu kolejkowego zamiast JMS API

# Formaty wiadomości

- *TextMessage* – przesyłanie obiektów java.lang.String
- *MapMessage* – mapa, zbiór obiektów w postaci nazwa-obiekt
- *BytesMessage* – format binarny, strumień w postaci bajtów
- *StreamMessage* – strumień typów prostych wpisywanych oraz czytanych sekwencyjnie
- *ObjectMessage* – przechowuje serializowalny obiekt Javy

# Interfejsy wykorzystywane przez klientów

- **MessageProducer** - tworzony przez obiekt sesji,służy do wysyłania wiadomości do Destination (Queue lub Topic):
  - QueueSender / TopicPublisher
  - Możliwość ustawienia DeliveryMode, priorytetu, czasu ważności
- **MessageConsumer** - tworzony przez obiekt sesji, służy do odbierania wiadomości
  - QueueReceiver / TopicSubscriber
  - synchroniczny lub asynchroniczny odbiór

# Interfejsy wykorzystywane przez klientów

- **MessageListener** – służy do nieblokującego odbierania komunikatów

  - Klasa która implementuje ten interfejs definiuje metodę onMessage()

  - Jej obiekt jest rejestrowany w konsumencie przez wywołanie metody setMessageListener
- **MessageSelector** - ”string” który pozwala na filtrowanie przychodzących komunikatów
  - bazuje na nagłówkach i właściwościach
  - nie filtruje treści
  - format podobny do zapytań SQL, np.:
    Login= ’Ala’ OR Login= ’Ela’

# Implementacja PTP

```
// context
Properties props = new Properties();
props.put(Context.INITIAL_CONTEXT_FACTORY,
        "org.exolab.jms.jndi.InitialContextFactory");
props.put(Context.PROVIDER_URL, "tcp://localhost:3035/");
Context context = new InitialContext(props);
System.out.println("Context OK");

// connection
QueueConnectionFactory factory =
        (QueueConnectionFactory)context.lookup("ConnectionFactory");
QueueConnection queueConnection = factory.createQueueConnection();
System.out.println("Connection OK");

// session & queue
QueueSession queueSession =
        queueConnection.createQueueSession(false,
            Session.AUTO_ACKNOWLEDGE);
Queue queue = (Queue)context.lookup("queue1");
System.out.println("Queue OK");
```

# Implementacja PTP - Producent

```
// PRODUCER
QueueSender sender = queueSession.createSender(queue);
queueConnection.start();
System.out.println("Connection started");

BufferedReader br =
        new BufferedReader(new InputStreamReader(System.in));

while(true){
    System.out.println("Type message: ");
    String msg = br.readLine();
    TextMessage message = queueSession.createTextMessage(msg);
    sender.send(message);
    System.out.println("Message sent");
}
```

# Implementacja PTP - Konsument

```
// CONSUMER
QueueReceiver receiver = queueSession.createReceiver(queue);
queueConnection.start();
System.out.println("Connection started");

while(true){
    System.out.println("Waiting for message...");
    TextMessage message = (TextMessage) receiver.receive();
    System.out.println("Received message: " + message.getText());
}
```

# Zadanie

- Zmodyfikować przykład:
  - wykorzystanie Topic zamiast Queue
  - Możliwość zarówno wysyłania jak i odbierania wiadomości
  - Asynchroniczny odbiór wiadomości (implementacja MessageListener)

# Zadanie domowe

- System aukcyjny
  - Kilka tematów jako kategorie aukcji
  - Uczestnik może:
    - Wystawić przedmiot
    - Śledzić daną kategorię
    - Przeprowadzić aukcję
  - Oferta zawiera:
    - Nazwę
    - Cenę
    - Opis słowny
    - Czas ważności oferty

# Zadanie domowe

- Wymagania:
  - Działanie na wielu hostach
  - Konfiguracja (IP, wpisanie z jakich tematów klient ma korzystać)
- Dodatkowo:
  - Durable subscriptions
  - Mechanizm expire
  - Filtrowanie
  - Priorytety
- Może być .Net, np.: Spring.NET

# Zadanie domowe

- Spakowany katalog JMS_2012_Nazwisko_Imie
- Mail z tematem tj. nazwa katalogu na adres **fmal@galaxy.agh.edu.pl**

- Linki:
  http://docs.oracle.com/javaee/5/tutorial/doc/bncdq.html
  http://openjms.sourceforge.net/index.html


AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

# Dziękuję