
ORACLE®

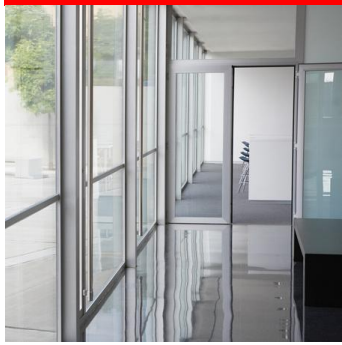
ORACLE

WebLogic JMS
Messaging für Fortgeschrittene

Marcel Amende
Business Unit Middleware

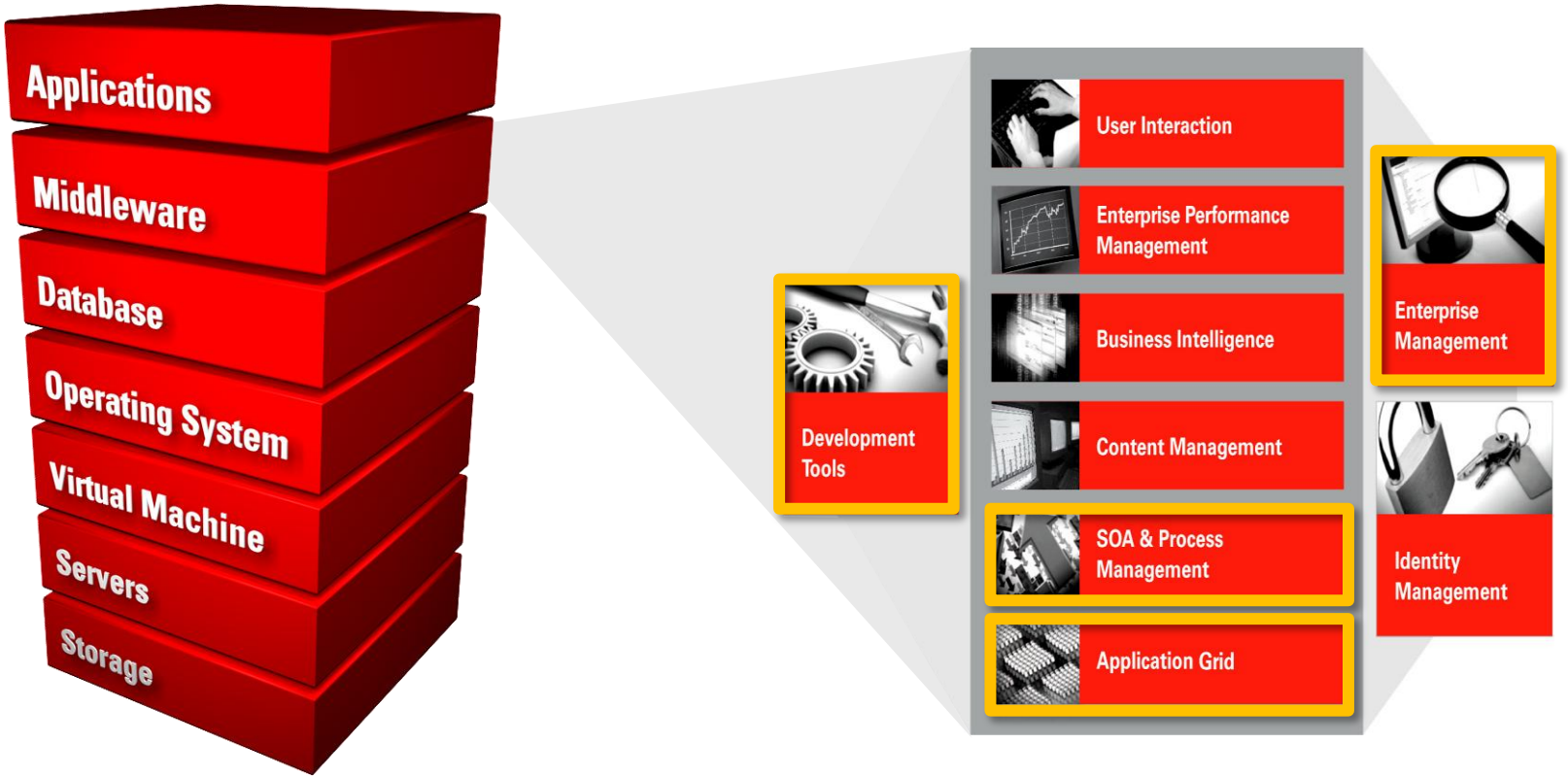
The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.

Agenda



- Weblogic JMS
- JMS Clustering
- JMS Unit-Of-Order
- JMS Unit-Of-Work
- JMS Store and Forward

Oracle Fusion Middleware

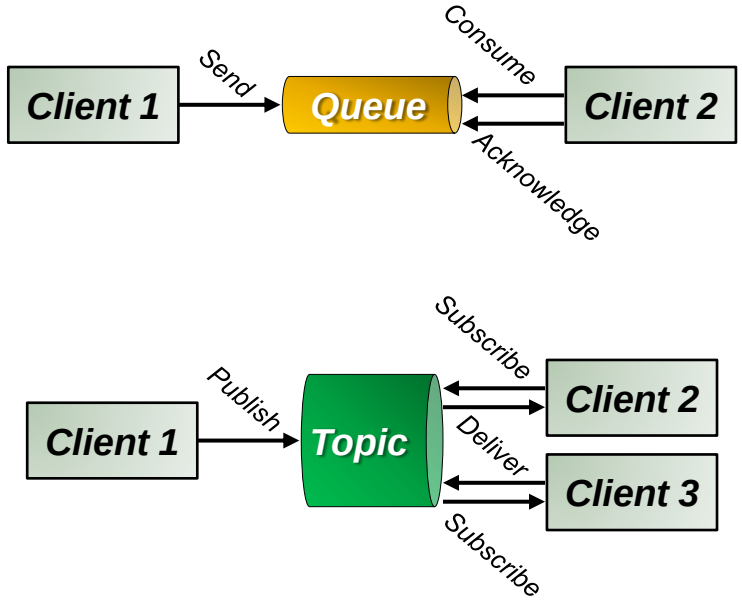


Java Messaging Service

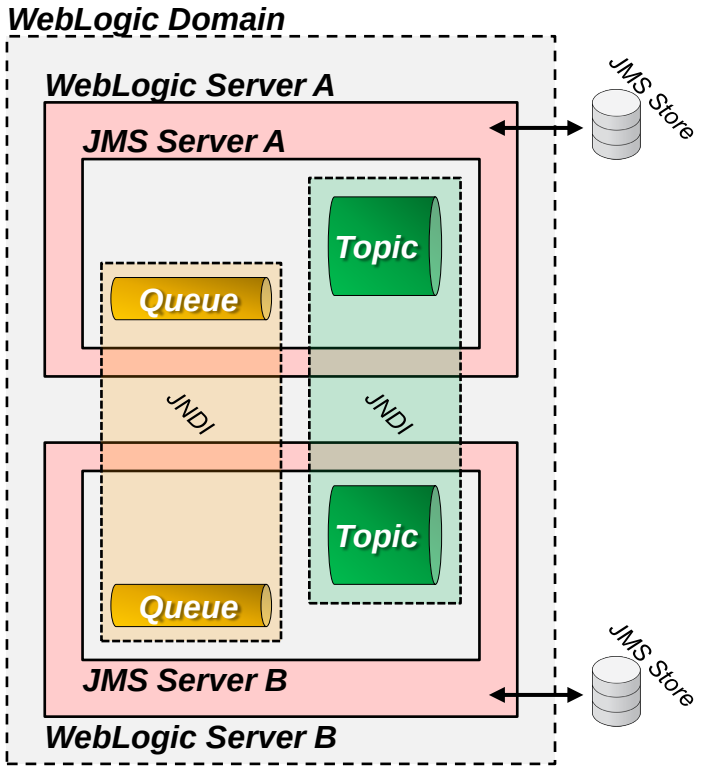
- API für Zugriff auf Enterprise Messaging Systeme
- Standard, Teil von JEE
- ermöglicht Java-Anwendungen Nachrichtenaustausch
- „Entkopplung“ von Produzent und Konsument
- Transportprotokoll (hier: T3, IIOP, HTTP), Komfort- und Clustering-Konzepte nicht standardisiert

Queue vs. Topic

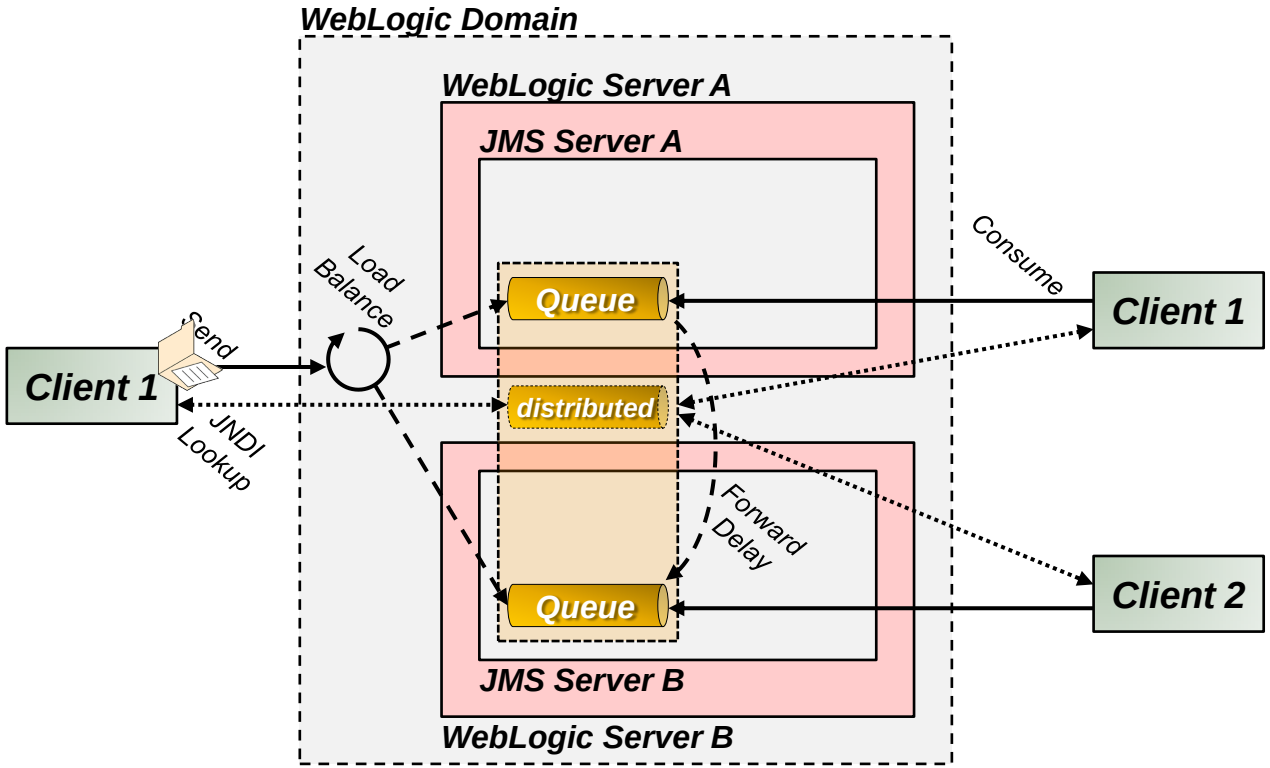
- Queue
  - Punkt-zu-Punkt
  - Lieferung an nur einen Konsumenten
- Topic
  - Publish-Subscribe
  - jeder passende Konsument erhält Nachricht



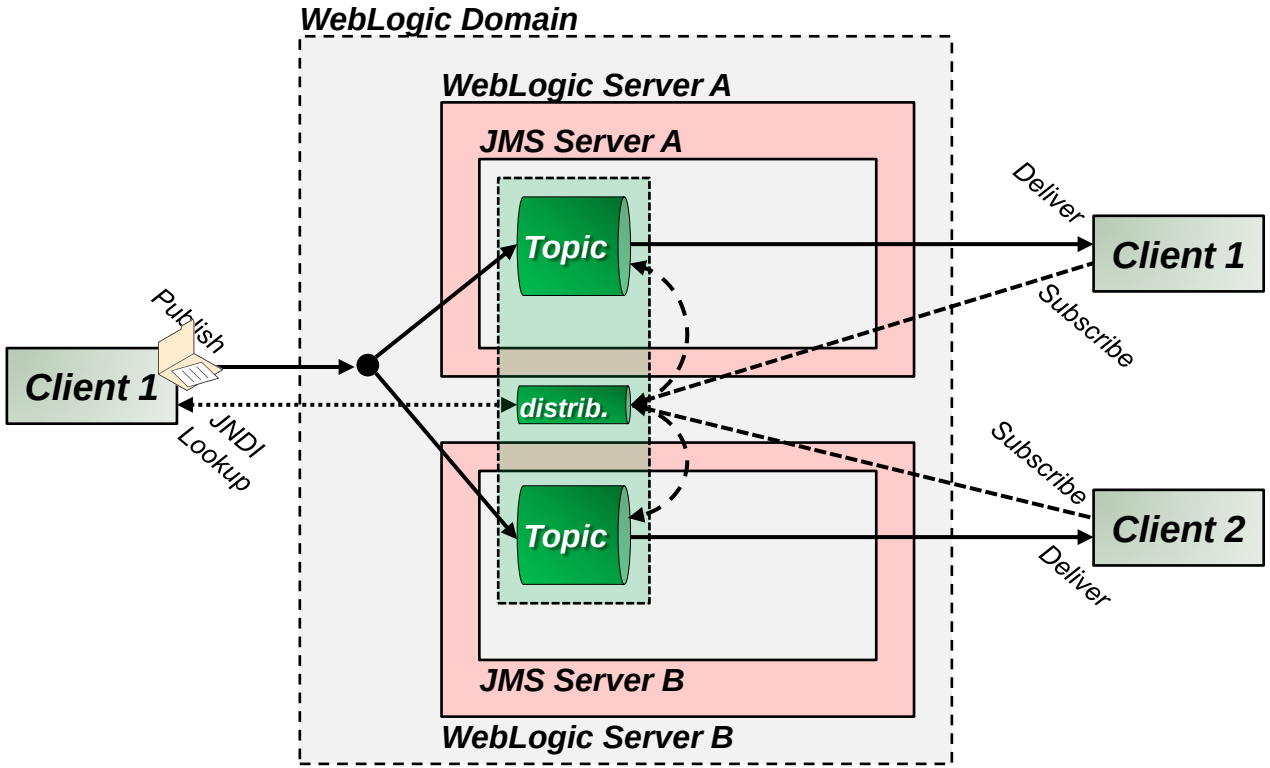
JMS Clustering



Distributed Queue



Distributed Topic



JMS Failure Recovery

- Server mit persistenten JMS Endpunkten (File oder JDBC) fällt aus
- Nachrichten sind „gestrandet“, da nur über Endpunkt erreichbar
- Lösungen:
  - Server Neustart
  - Server Neustart auf einer anderen Maschine
    (Whole Server Migration, „Lease“-basiert, erster Server ist „Master“)
  - JMS Server Neustart an anderem Ort (Service Migration)
    - Rettung „gestrandeter“ Nachrichten

Problem: Nachrichtenreihenfolge

Thema „Nachrichtenreihenfolge“:

- in JMS Spezifikation nur grundlegend behandelt
- 1-zu-1: zwischen einem Produzenten und einem Konsumenten

Was macht man bei...

? mehreren Produzenten

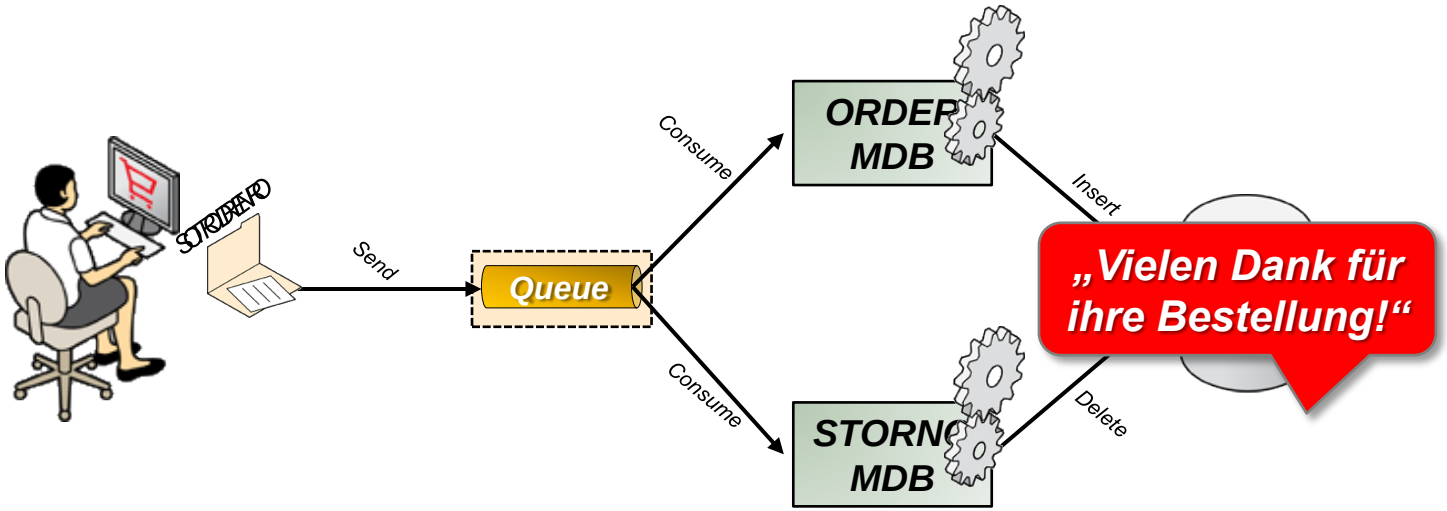
? mehreren Konsumenten

? Nachrichtenwiederherstellung

? zurückrollenden Transaktionen

? eigenen Sortierschlüsseln

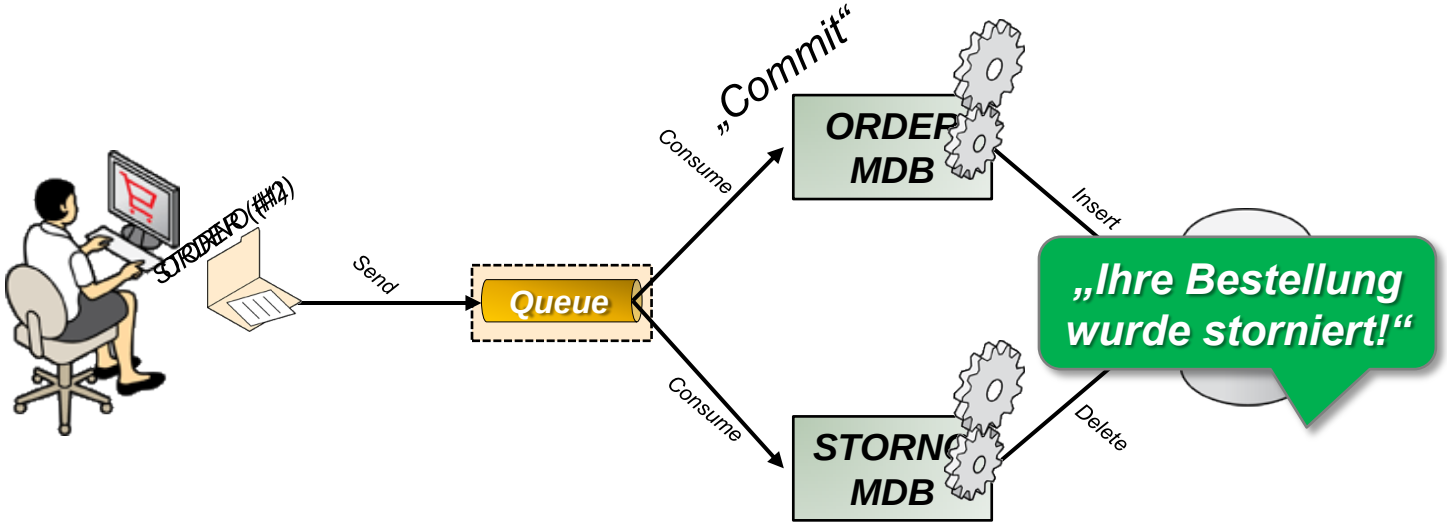
Negativbeispiel



Lösung: Unit-of-Order

- gruppierung von Nachrichten eines oder mehrerer Produzenten in eine Einheit („Unit“)
- streng sequentielle Abarbeitungsreihenfolge
- Blockierung einer Nachricht bis die Verarbeitung der vorhergehenden bestätigt ist:
  - „Acknowledged“
  - „Commited“
  - „Rolled Back“
  - „Recovered“
- Distributed: alle Nachrichten einer Einheit an denselben Endpunkt

Beispiel mit „Unit-of-Order“



Unit-of-Work

- noch striktere Sicht auf Nachrichtengruppen
- JMS Nachrichten werden als Gruppe von einem Konsumenten verarbeitet
- Produzent definiert Satz von Nachrichten
- unterbrechungsfreie Auslieferung

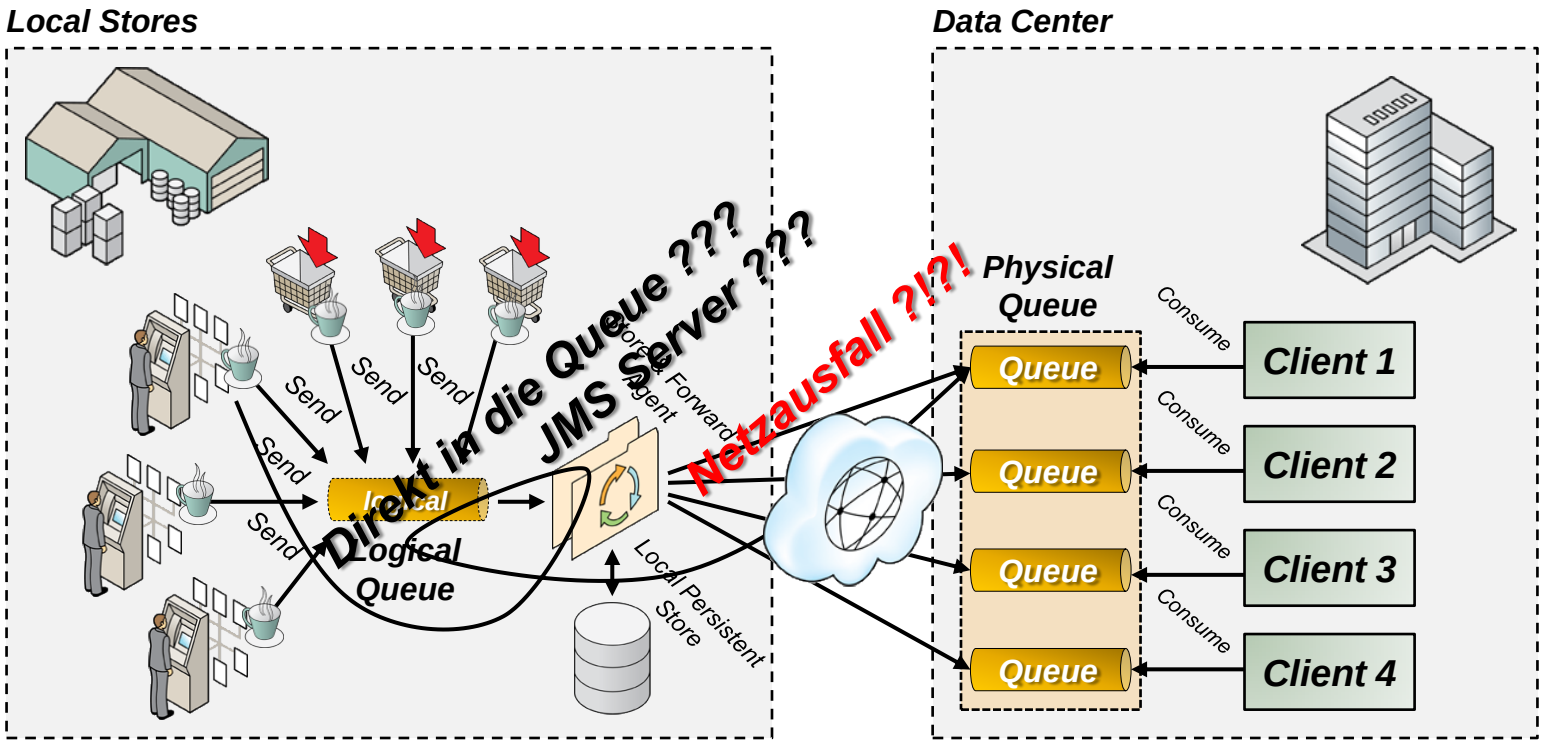
Codebeispiel „Produzent“

```
for(int i=1; i<=100; i++){
  sendMsg.setStringProperty("UnitOfWork", "My Unit");
  sendMsg.setIntProperty("UnitOfWorkSequenceNumber", i);
  if(i == 100){
    System.out.println("Endkennzeichen: " + i);
    sendMsg.setBooleanProperty("IsUnitOfWorkEnd", true);
  }
  qSender.send(sendMsg, DeliveryMode.PERSISTENT, 7, 0);
}
```

Codebeispiel „Konsument“

```
msg = qReceiver.receive();
System.out.println("Empfangene Nachrichtengruppe: " + msg);
ArrayList msgList =
   (ArrayList)(((ObjectMessage)msg).getObject());
num = msgList.size();
System.out.println("Anzahl Nachrichten in Gruppe: " + num);
```

Store-and-Forward



Q&A

Hardware and Software

ORACLE

Engineered to Work Together


ORACLE®