Konkrete Techniken für asynchrone Kommunikation

Konkrete Techniken - Ziele

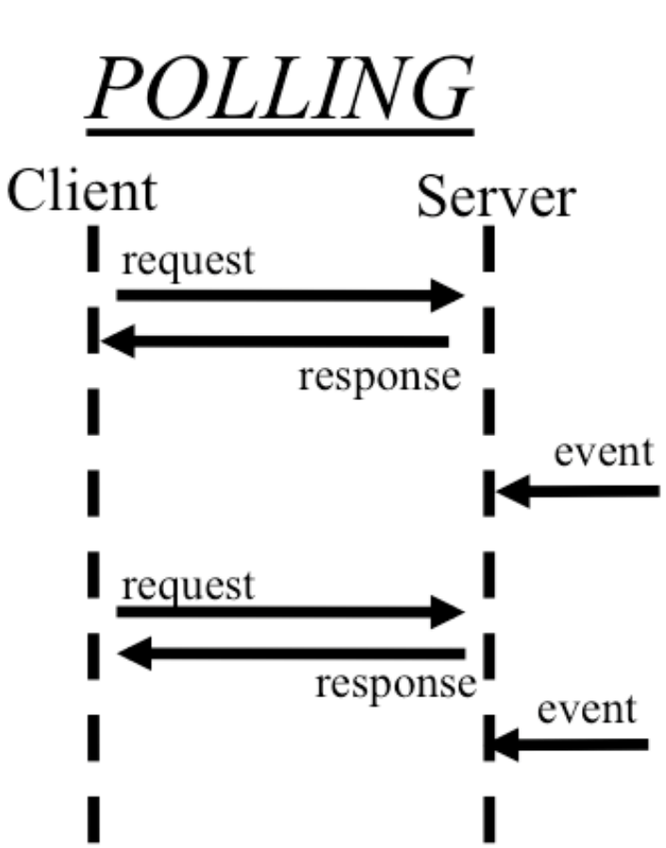
- Techniken und Bibliotheken für die asynchrone Client/Server Kommunikation im Web einordnen können und im eigenen Projekt Kontext anwenden können.
- Produkte, Protokoll und Standards für Message Oriented Middleware einordnen können.
- Cloud Services für die asynchrone Kommunikation nennen und einordnen können.

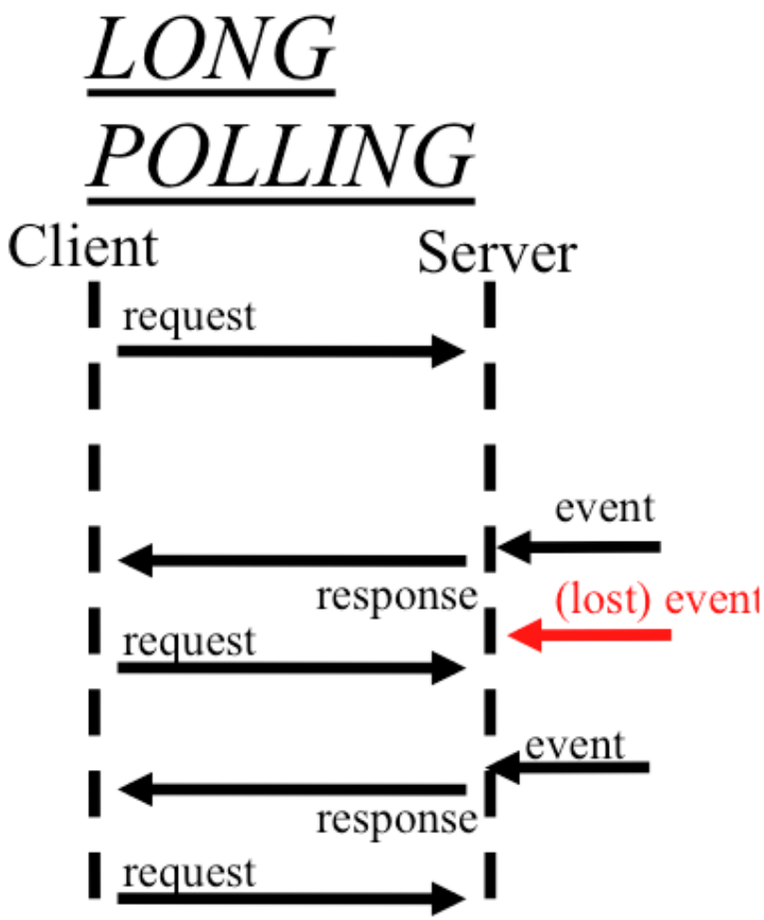
Pull vs. Push

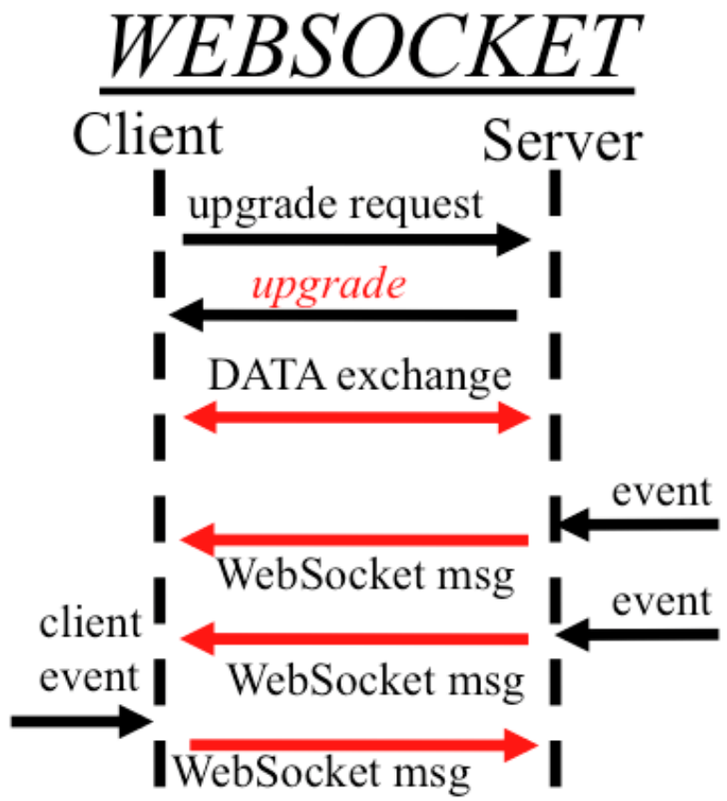
- Poll (Abfragen) und Pull (Ziehen)
- Regelmäßiges Abfragen von Daten
- Initiiert vom Client
- Viele Verbindungen
- Bei vielen Änderungen
- Daten müssen nicht synchron gehalten werden
  oder sind nicht zeitkritisch
  - Wetter, Dropbox

- Push (drücken, stoßen, schieben)
- Echtzeit Kommunikation
- Initiiert vom Client, ausgelöst vom Server
- dauerhafte Verbindung(en)
- Immer wenn es „zeitkritisch“ ist
- Bei ereignisorientierten Anwendungsfällen

Asynchronität mit HTTP







Websocket Upgrade request

```
 GET ws://echo.websocket.org/?encoding=text HTTP/1.1

Origin: http://websocket.org

Cookie: __utma=99as

Connection: Upgrade

 Host: echo.websocket.org

Sec-WebSocket-Key: uRovscZjNol/umbTt5uKmw==

Upgrade: websocket

Sec-WebSocket-Version: 13
```

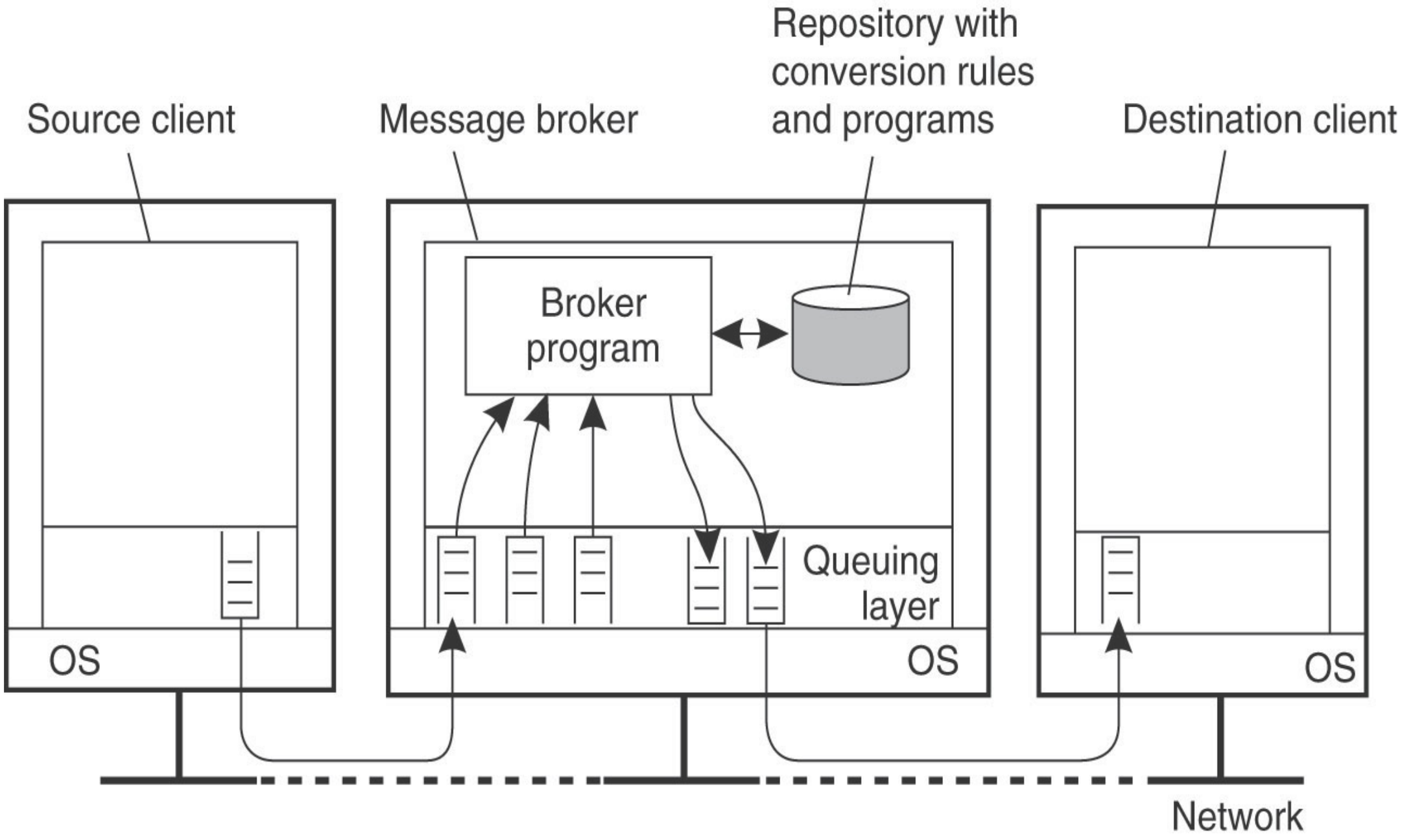
Websocket upgrade response

```
HTTP/1.1 101 WebSocket Protocol Handshake

Date: Fri, 10 Feb 2012 17:38:18 GMT

Connection: Upgrade

Server: Kaazing Gateway

Upgrade: WebSocket

Access-Control-Allow-Origin: http://websocket.org

Access-Control-Allow-Credentials: true

Sec-WebSocket-Accept: rLHCkw/SKsO9GAH/ZSFhBATDKrU=

                    Access-Control-Allow-Headers: content-type
```

Asynchrone C/S Kommunikation in js

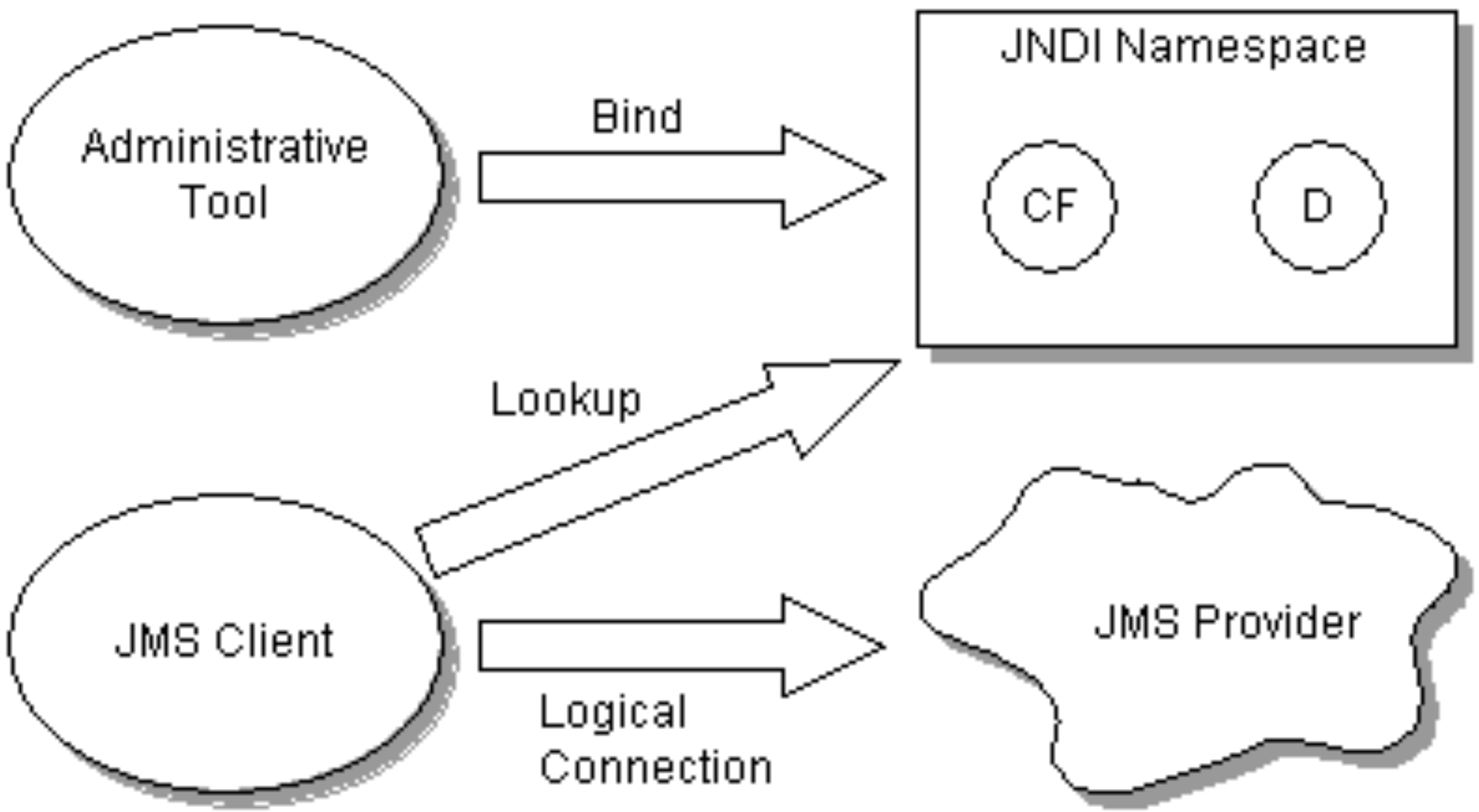
- socket.io: framework, das verschiedene http basierte asynchrone Kommunikationsmethoden zwischen Client und Server aushandelt.
- faye: publish/subscribe messaging ursprünglich für Browser basierte Applikationen; wird aber auch als generelle Middleware verwendet
- Bayeux: Das Publish/subscribe Protokoll, das unter anderem von faye benutzt wird

Message Broker



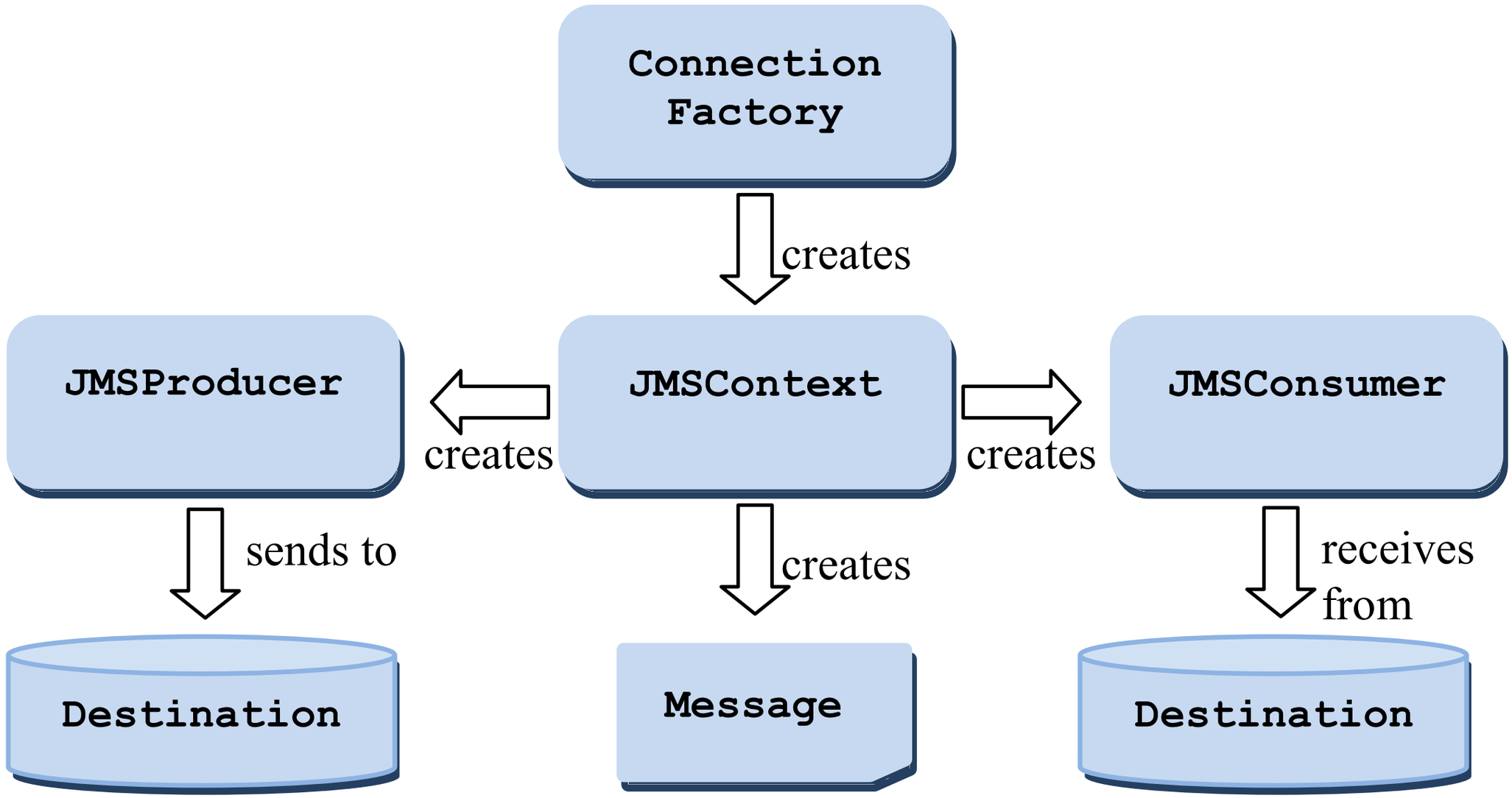
Quelle: Tanenbaum et al.: Distributed Systems

JMS API Architektur



Quelle: http://download.oracle.com/otn-pub/jcp/jms-2_0-fr-spec/JMS20.pdf, Abruf 3.5.17

JMS 2.0 Generic (simplified) API



Quelle: http://download.oracle.com/otn-pub/jcp/jms-2_0-fr-spec/JMS20.pdf, Abruf 3.5.17

Systeme, Protokolle, Standards

- ActiveMQ (Apache Foundation, JMS Provider)
- RabbitMQ (VMware (Open Source)
- Application Sever wie Glassfish (Oracle (Open Source))
- WebSphere/MQ (IBM)
- XMPP Server (siehe: https://xmpp.org/software/servers.html)
- **J**ava **M**essaging **S**ervices (Teil der JEE Specs)
- **A**dvanced **M**essage **Q**ueuing **P**rotocol (ISO/IEC 19464)
- E**X**tensible **M**essaging and **P**resence **P**rotocol (RFC 6120, RFC 6121)
- **M**essage **Q**ueue **T**elemetry **T**ransport (OASIS Standard, Eclipse Projekt)

Cloud Dienste

- Amazon Web Services (AWS): SQS, SNS
- RabbitMQ in the Cloud: z.B. https://www.rabbitmqhosting.com, https://bitnami.com/stack/rabbitmq
- XMPP (diverse hosting Anbieter)
- Apple Push Notifications (APN)
- Google firebase

Konkrete Techniken - Zusammenfassung

- Asynchronität am Front End
- Message Broker - Produkte und Standards
- Cloud Services