Web-Engineering

Web-Sockets:
bi-direktionale Kommunikation im Web

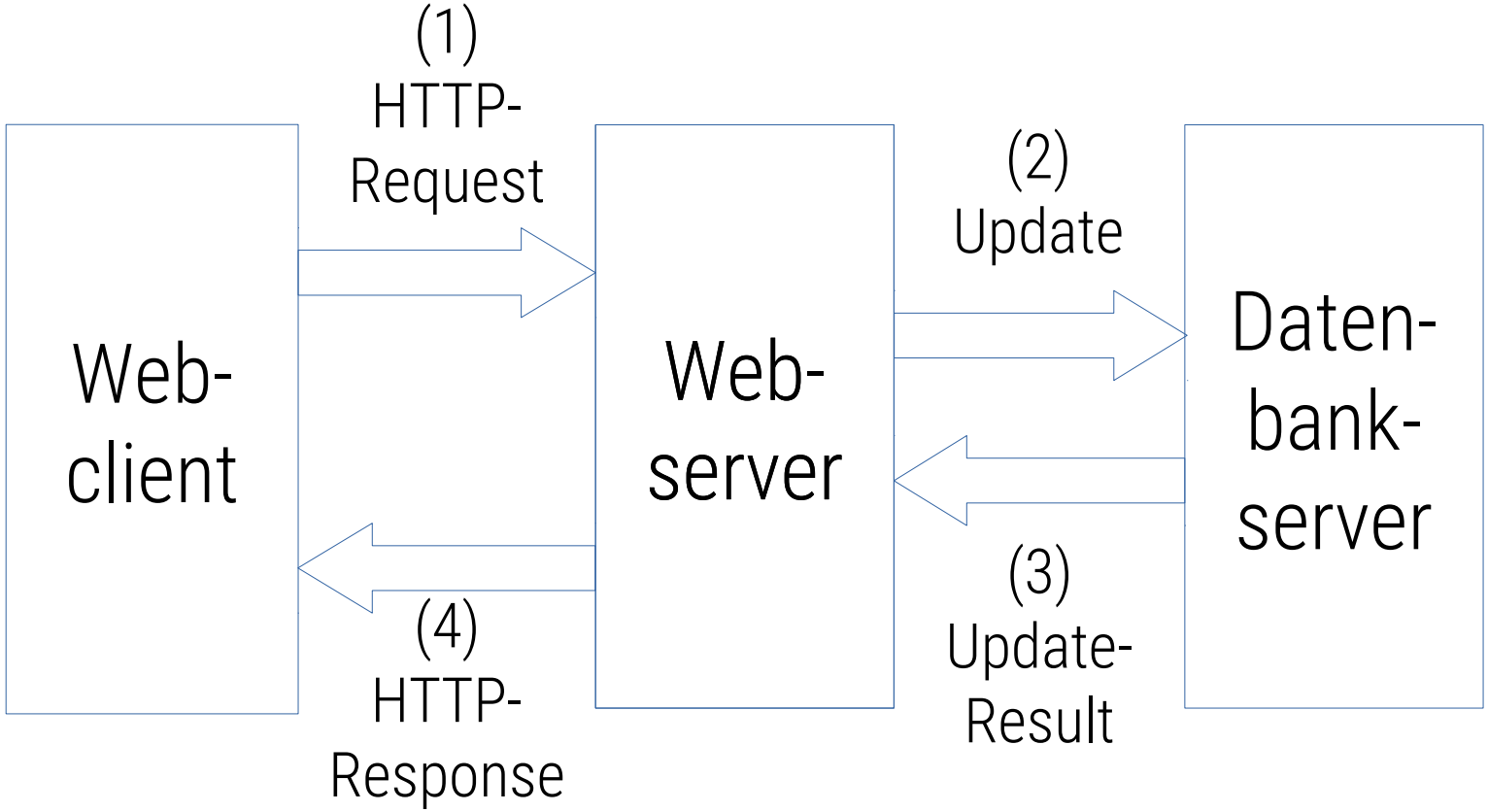
Hochschule Niederrhein
University of Applied Sciences
Elektrotechnik und Informatik
Faculty of Engineering and Computer Science

Web-Anwendungen mit HTTP (1)

- Client-Server-Architektur
- HTTP: Hypertext-Transfer-Protocol
  - Zustandslos
  - Request: Client → Server
  - Response: Server → Client
- Probleme:
  - Aktualisierungen beim Client nur aufgrund von Requests
  - Aktualisierungen anderer Clients nicht direkt möglich

Web-Anwendungen mit HTTP (2)

Ablauf einer Änderung in der Datenbasis:



Web-Anwendungen mit HTTP (3)

- Reihenfolge der Nachrichten / Operationen
  1. HTTP-Request: Anfrage des Client an den Server
  2. Update: Änderungsoperation aufrufen
  3. Update-Result: Ergebnis der Änderungsoperation auswerten
  4. HTTP-Response: Ergebnis an den Client ausliefern

Wie erfahren andere Clients von der Änderung?

Web-Anwendungen mit HTTP (4)

- Change-Propagation:
  - Hinweis auf Änderungen in einem System verteilen
- Varianten bei „klassischen“ Web-Anwendungen mit HTTP:
  - Polling:
    - die Client-Systeme fragen regelmäßig an (z.B. alle 10 Sekunden), ob es Änderungen gibt
    - Erzeugt viele nutzlose Anfragen
    - Information steht ggf. erst nach Ablauf der Wartezeit zur Verfügung
  - Long-Running-Requests:
    - Response zurückhalten bis Änderungen vorliegen
    - Viele offene Anfragen / gleichzeitige Verbindungen

Web-Anwendungen mit HTTP (5)

- Weitere Varianten:
  - SSE (Server-Sent Events):
    - Auslieferung von Nachrichten vom Server zum Client zu einem beliebigen Zeitpunkt
    - Wird aber vom Client eingerichtet, also „erwartet“
    - `EventSource`-Interface
      - Anfrage beim Server einrichten
      - Message-Listener zum asynchronen Empfang
    - (siehe: https://developer.mozilla.org/en-US/docs/Web/API/Server-sent_events/Using_server-sent_events)

Web-Anwendungen mit HTTP (6)

- HTTP2: Generelle Verbesserungen beim Durchsatz
  - Anzahl Anfragen (Round-Trips) verringern
  - Nebenläufigkeit verbessern
- HTTP2 Server-Push:
  - z.B. beim Laden von Webseiten gleichzeitige Auslieferung mehrerer Ressourcen (z.B. Markup, CSS)
  - Somit eher zur Durchsatzverbesserung, nicht für Change-Propagation gedacht
- PUSH-Interface: experimentell, nicht standardisiert

Web-Sockets (1)

Ziel: Bi-Direktionale Verbindungen ermöglichen

- Bestandteil der Web-Standards im Rahmen der „HTML5“-Familie
- Standardisiertes Protokoll seit etwa 2011
- Zunächst HTTP-Anfrage, mit der dann die Umschaltung auf das Web-Socket-Protokoll vorgenommen wird
- Effektive Unterstützung durch Webbrowser seit etwa 2012 (IE 10, FF 11, Chrome 16, Safari 6)
- W3C-Socket-API definiert die Verwendung

Web-Sockets (2)

W3C-Web-Socket-API (Application Programming Interface):

- Web-Socket-Objekt (Bsp: javascript)

```
var ws_o = new WebSocket('ws://localhost')
```

- Zustände:

  CONNECTING → Das Objekt versucht eine Verbindung zur angegebenen URL herzustellen.

  OPEN → Die Verbindung wurde erfolgreich hergestellt. Nachrichten können empfangen und gesendet werden.

  CLOSING → Die Verbindung wird geschlossen.

  CLOSED → Die Verbindung ist geschlossen. Die Verbindung wurde beendet oder konnte nicht geöffnet werden.

Web-Sockets (2)

W3C-Web-Socket-API (Application Programming Interface):

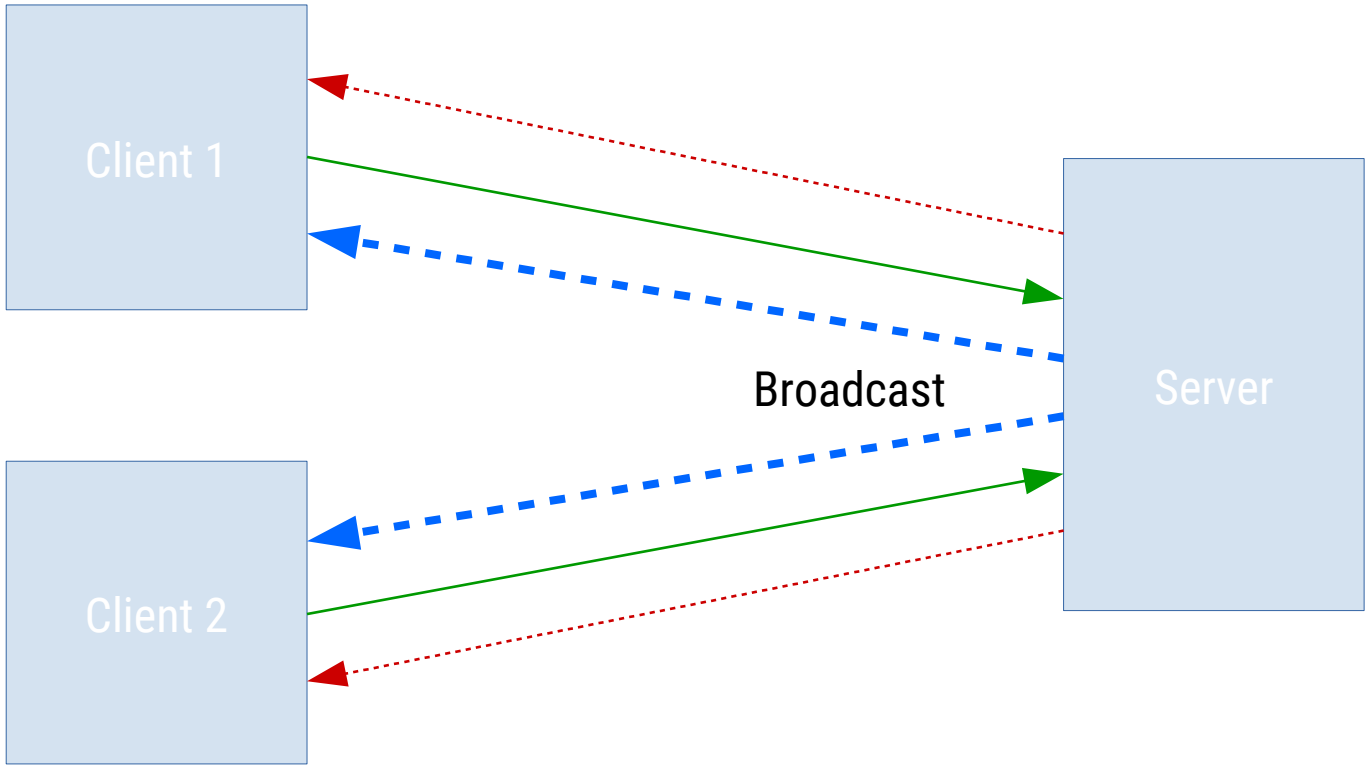
- Ereignisse:
  - onopen → Verbindung hergestellt
  - onerror → Verbindungsfehler
  - onclose → Verbindung wurde geschlossen
  - onmessage → Daten empfangen
- Senden von Daten
  - An einzelnen Client
  - Broadcast

Beispiel (1)

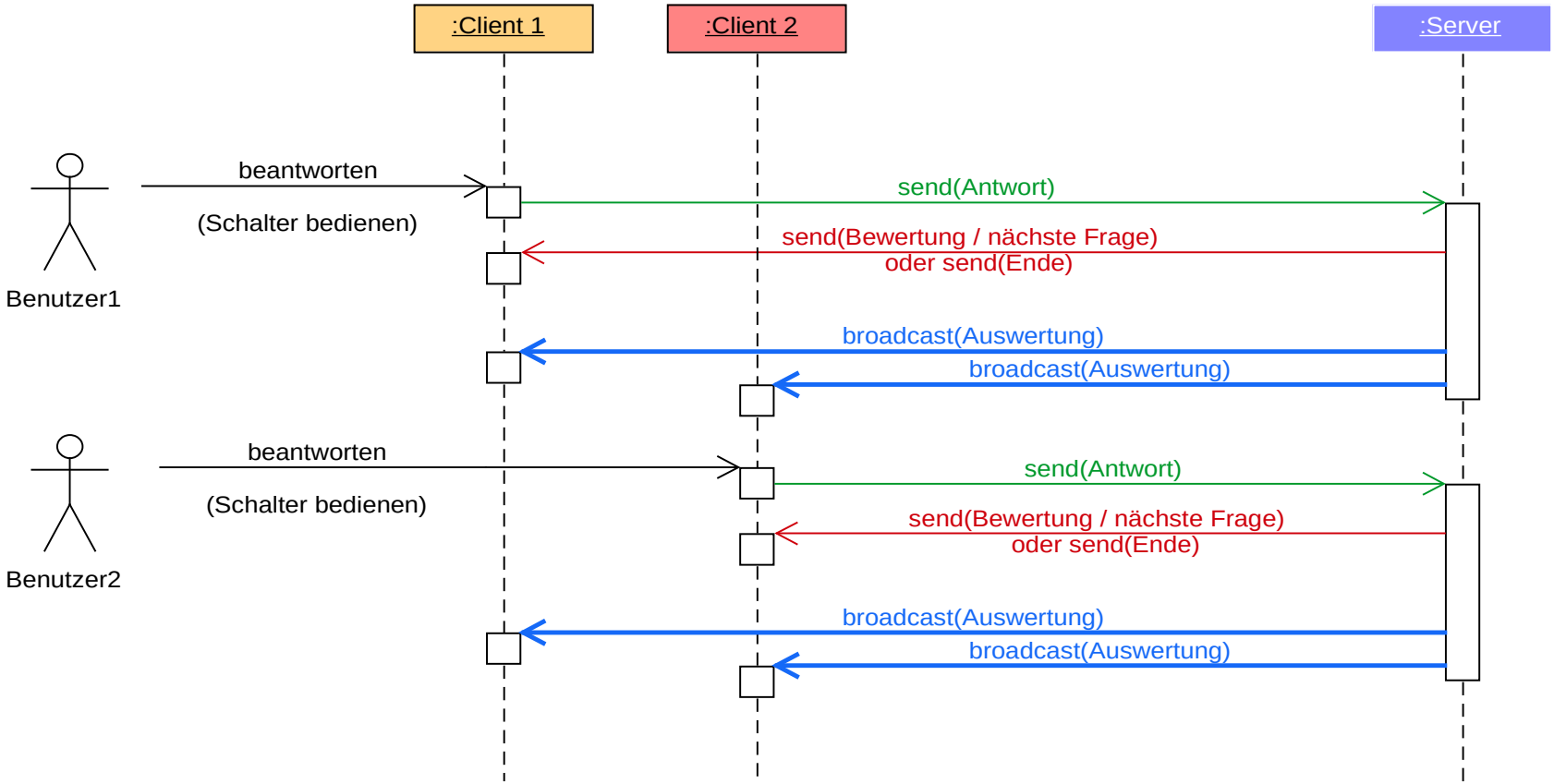
Quiz-System

- N Clients:
  - Erhalten Fragen / antworten mit Ja oder Nein
  - Erhalten Rückmeldung
  - Erhalten *aktuelle* Informationen zur Beantwortung durch alle Teilnehmer
- Server:
  - Verwaltet alle Anfragen
  - Informiert per „Broadcast“ alle Teilnehmer

Beispiel (2)



Beispiel (3)


:Client 1
:Client 2
:Server
Benutzer1
beantworten
(Schalter bedienen)
send(Antwort)
send(Bewertung / nächste Frage)
oder send(Ende)
broadcast(Auswertung)
broadcast(Auswertung)
Benutzer2
beantworten
(Schalter bedienen)
send(Antwort)
send(Bewertung / nächste Frage)
oder send(Ende)
broadcast(Auswertung)
broadcast(Auswertung)