UNIVERSITÀ DI TORINO

Http: Streaming API

Prof. Fabio Ciravegna
Dipartimento di Informatica
Università di Torino
fabio.ciravegna@unito.it

HTTP/1.1 Enhancements

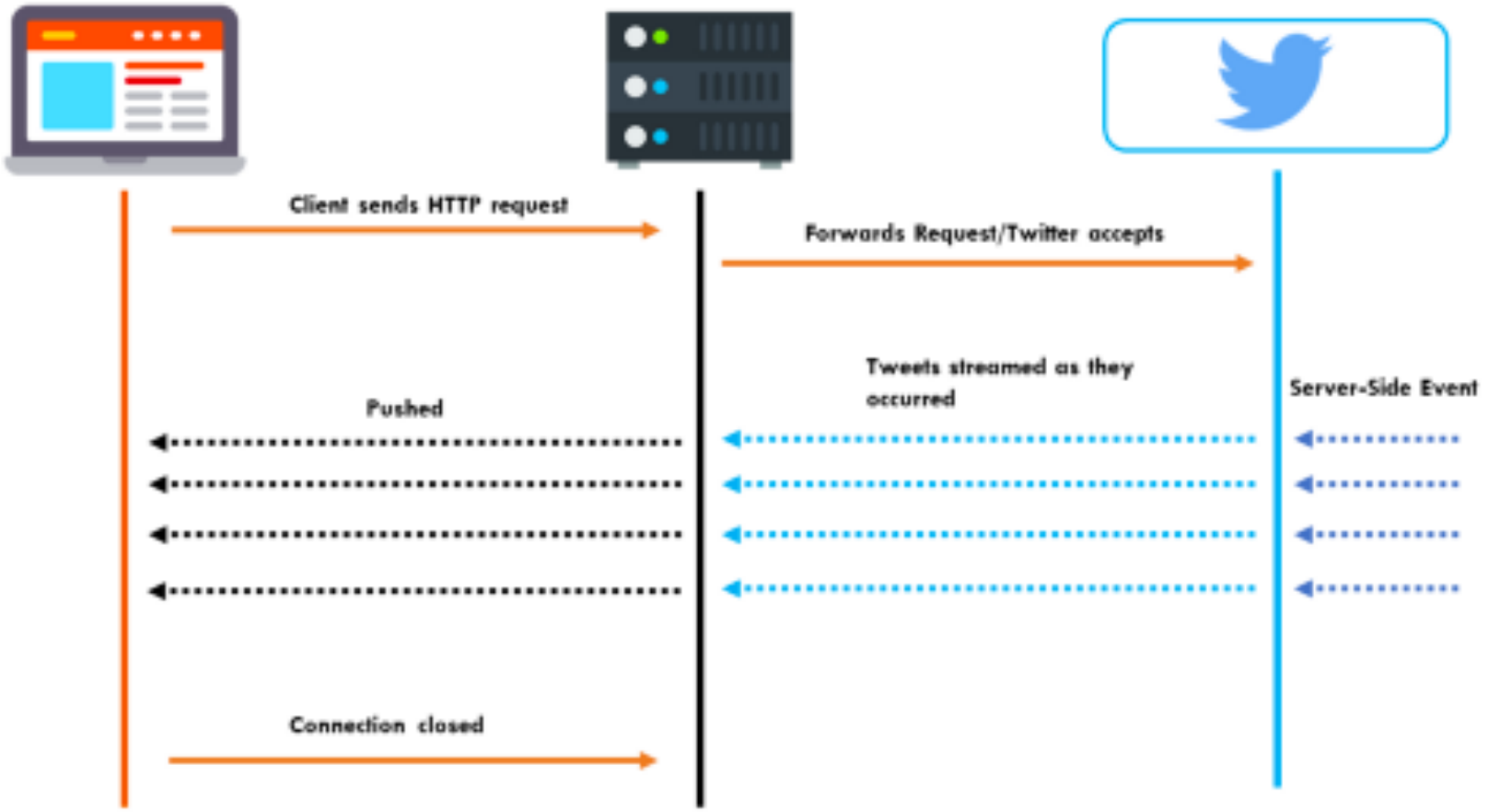
https://medium.com/platform-engineer/web-api-design-35df8167460

- It implementa critical performance optimisations and feature enhancements, e.g.
  - persistent and pipelined connections,
  - chunked transfers,
  - new header fields in request/response body etc.
- Two main headers
  - most of the modern improvements to HTTP rely on these two:
    - Keep-Alive
      - to set policies for long-lived communications between hosts (timeout period and maximum request count to handle per connection)
    - Upgrade
      - to switch the connection to an enhanced protocol mode such as HTTP/2.0 (h2,h2c) or Websockets (websocket)
-

Client-server communication

- RESTful APIs:
  - the most common APIs: the client requests a service and the server satisfies it
    - request/response model
- HTTP Polling
  - the client polls the server requesting new information
    - HTTP Short Polling: practically never used as it procures intense traffic
    - HTTP Long Polling: rather expensive but used
    - HTTP Periodic Polling: With a predefined time gap between two requests
    - HTTP Streaming: the most used, provides a long-lived connection for instant and continuous data push
      - The server trickles out a response of indefinite length (it's like polling infinitely).
      - HTTP streaming is performant, easy to consume and can be an alternative to WebSockets

HTTP Streaming example: Twitter



https://medium.com/platform-engine

Streaming in Node/MySQL communication

```
var mysql = require('mysql');

var connection = mysql.createConnection(
    {
      host     : 'mysql_host',
      user     : 'your-username',
      password : 'your-password',
      database : 'database_name',
    }
);
connection.connect();
var query = connection.query('SELECT * FROM your_relation');

query.on('error', function(err) {
    throw err;
});

query.on('fields', function(fields) {
    console.log(fields);
});

query.on('result', function(row) {
    console.log('name: ' + row.name +
            ' ', row.surName);
});

query.on('end', function() {

// When it's done I Start something else
});
connection.end();
```

event received while processing: error

the list of fields in the next record

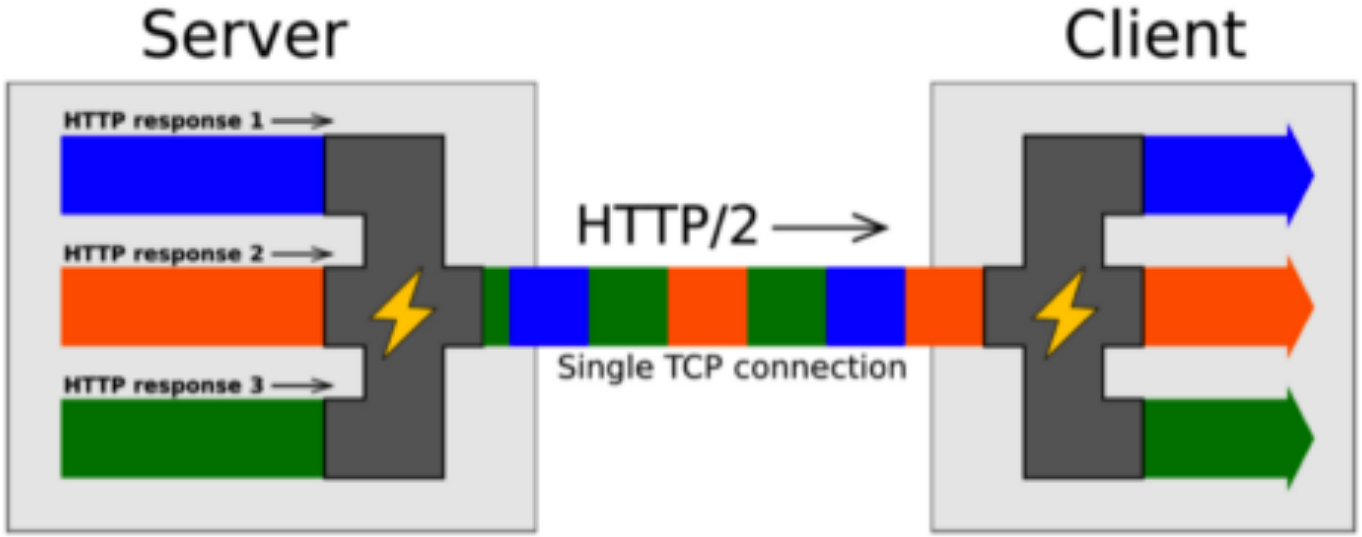
event received while processing: a row of data is available for processing
use elements from the fields variable to access parts of the row

when all rows have been received

HTTP/2 Server Push

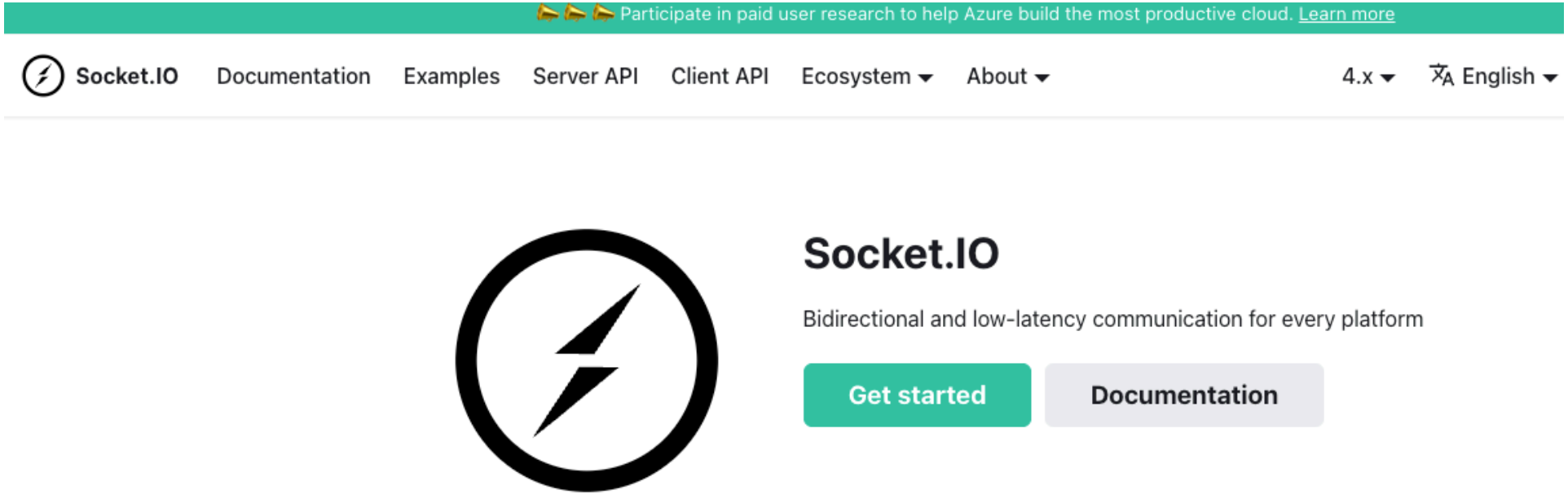
https://medium.com/platform-engineer/web-api-design-35df8167460

- A mechanism for a server to proactively push assets (stylesheets, scripts, media) to the client cache in advance



Finally: WebSocket

- The client / server communication is
  - bidirectional
  - and can be initiated by either components



UNIVERSITÀ DI TORINO

Questions?

© Prof. Fabio Cravegna, Università di Torino