/EXPRESS.JS

"You think, you can"

INTERNET COMMUNICATION

AFP

BitTorrent

HTTPS

TCP

POP

9P

IMAP

BGP

HTTP

UDP

SCP

SSL

ICMP

FTP

SSH

IP

PPP

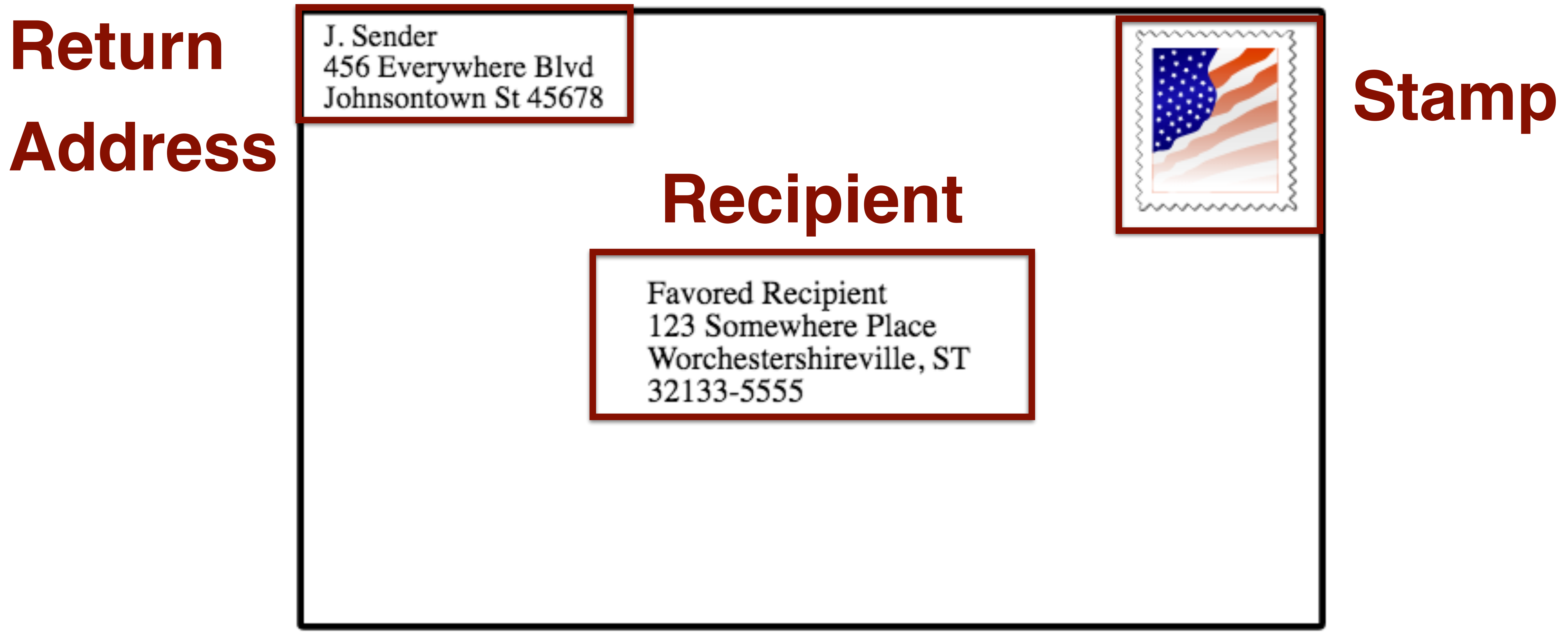
NFS

SMTP

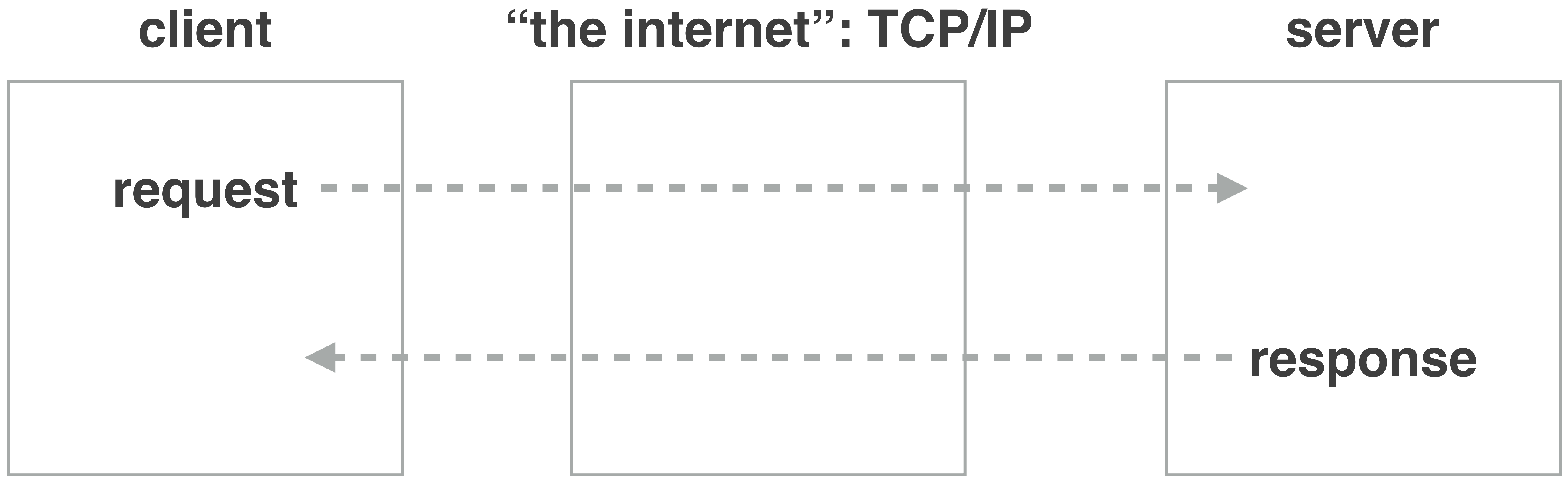
TCP/IP - MAIL ROOM

HTTP - ENVELOPE FORMAT

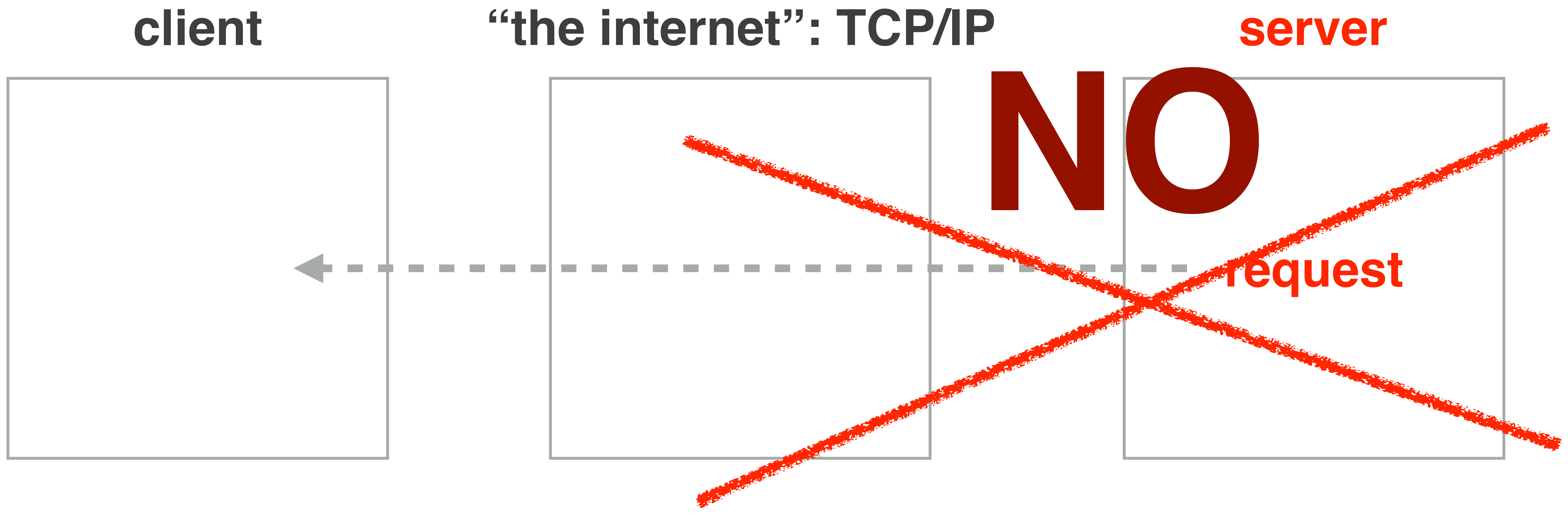
- Format for reliable, mirrored communication



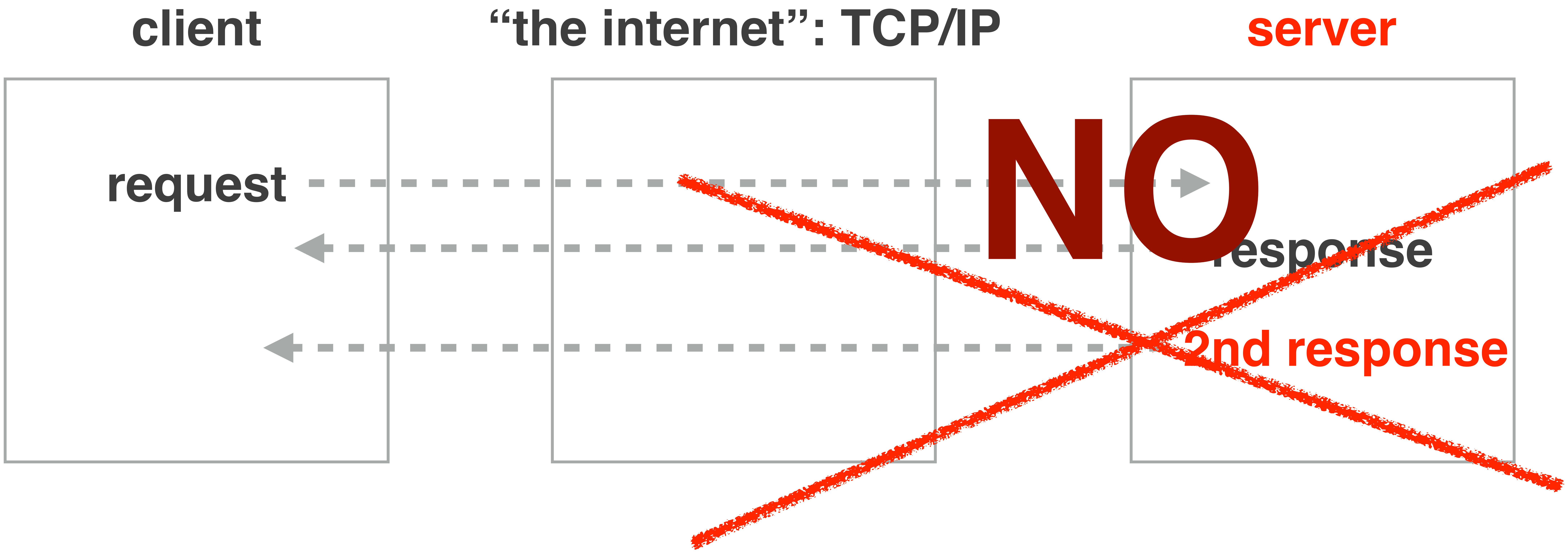
HTTP



HTTP



HTTP


client
"the internet": TCP/IP
server
request
response
2nd response
NO

PROTOCOL

- Specification NOT implementation
- Often used for communication

node's `http` library is an implementation of HTTP

HTTP REQUEST

just a message with a certain format

verb URI

headers

```
POST /docs/1/related HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)

bookId=12345&author=Nimit
```

body

(from http://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html)

COMMON VERBS

POST	“Create”
GET	“Read”
PUT	“Update”
DELETE	“Delete”

HTTP RESPONSE

status

headers

```
HTTP/1.1 200 OK
Date: Sun, 18 Oct 2009 08:56:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Sat, 20 Nov 2004 07:16:26 GMT
ETag: "10000000565a5-2c-3e94b66c2e680"
Accept-Ranges: bytes
Content-Length: 44
Connection: close
Content-Type: text/html
X-Pad: avoid browser bug

<html><body><h1>It works!</h1></body></html>
```

(from http://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html)

payload/body

COMMON STATUSES

200	“OK”
201	“created”
304	“cached”
400	“bad request”
401	“unauthorized”
404	“not found”
500	“server error”

EXPRESS

A node library for request handling

<CODE>

</CODE>

EXPRESS

- Treats requests as objects, created by event
- Matches on verb AND route
- Allows chaining of many handlers
- Enables modular layering with “routers”

POP QUIZ

CLIENT

Something that makes (HTTP) requests

SERVER

Something that responds to (HTTP) requests

REQUEST

A formatted message sent over the network by a client. Contains VERB, URI (route), headers, and body.

RESPONSE

A server's reply to a request (formatted message). Contains headers, payload, and status.

REQUEST-RESPONSE CYCLE

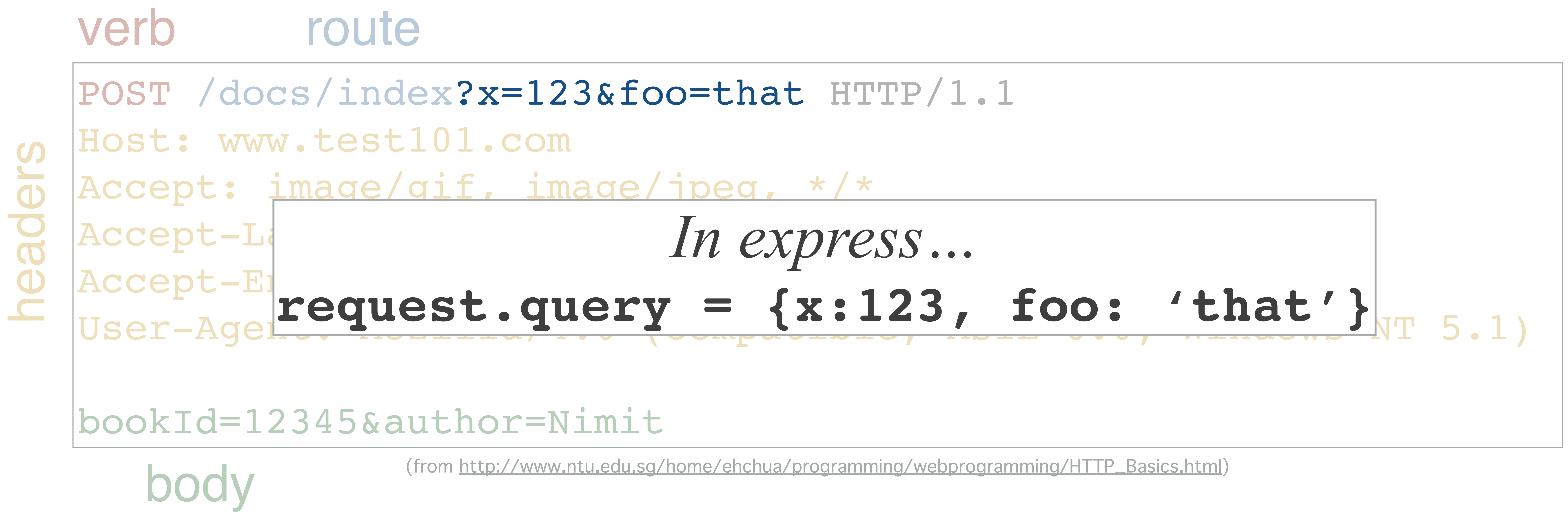
The client ***always*** *initiates by sending a request, and the server completes it by sending* ***exactly one*** *response*

EXPRESS MIDDLEWARE

A function that handles responding to requests.
Of the form: **`function(req, res, next){…}`**

REQUEST QUERY STRING

A way to pass data from client to server.



```
POST /docs/index?x=123&foo=that HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Accept-La
Accept-En
User-Agen                                          NT 5.1)

bookId=12345&author=Nimit
```

In express…

```
request.query = {x:123, foo: ‘that’}
```

(from http://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html)

REQUEST BODY

A way to pass data from client to server.

verb route

```
POST /docs/index?x=123&foo=that HTTP/1.1
Host: www.test101.com
Accept: image/gif, image/jpeg, */*
Ac
Ac
Us                                                          )

bookId=12345&author=Nimit
```

headers

body

In express…

```
request.body = {bookId:12345, author: ‘Nimit’}
```

(from http://www.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html)

REQUEST PARAMS

A variable portion of the URI.

ROUTER

A “layer” of route handlers (middlewares).

REMEMBER!!

- routes are not file paths
- order matters
- `req.params` v `req.query` v `req.body`
- `app.use` v `app.all`