



Corso di Progettazione di Applicazioni Web e Mobile

Hello!

I am Diego Bonura

Mi occupo di:

- Frontend
- Backend
- Mobile
- IoT
- Ricerca e sviluppo

diego@bonura.dev

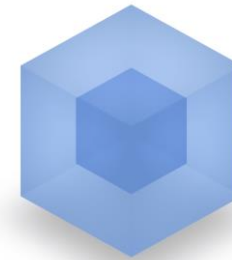
<https://medium.com/@diegobonura>



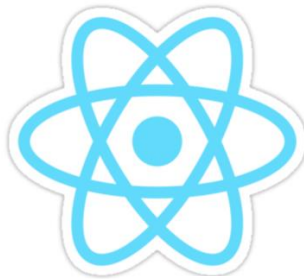
LOCCIONI



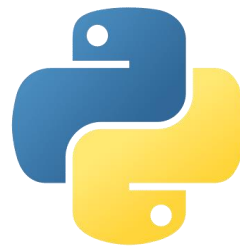
Cosa riconoscete?



Express



Apache





Programma

- Sviluppo web/mobile: di cosa si tratta
- Architettura di una applicazione mobile
- Protocolli
- Dalla prototipazione al deploy
- Sicurezza
- Testing
- ...



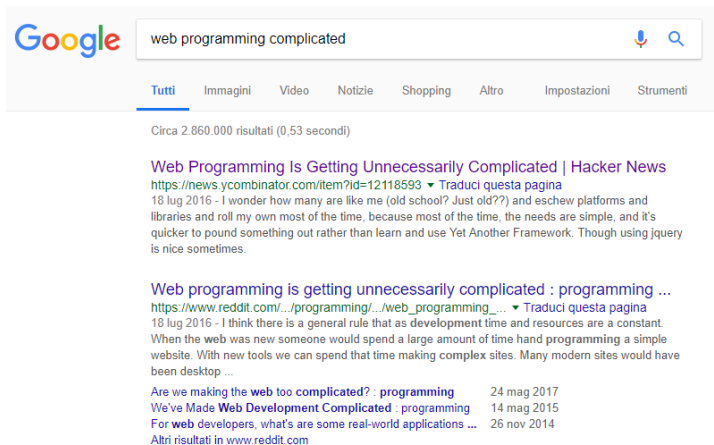
1.



Web e Mobile development



“


Quanto è complesso sviluppare applicazioni web/mobile?




Google  

[Tutti](#) [Immagini](#) [Video](#) [Notizie](#) [Shopping](#) [Altro](#) [Impostazioni](#) [Strumenti](#)

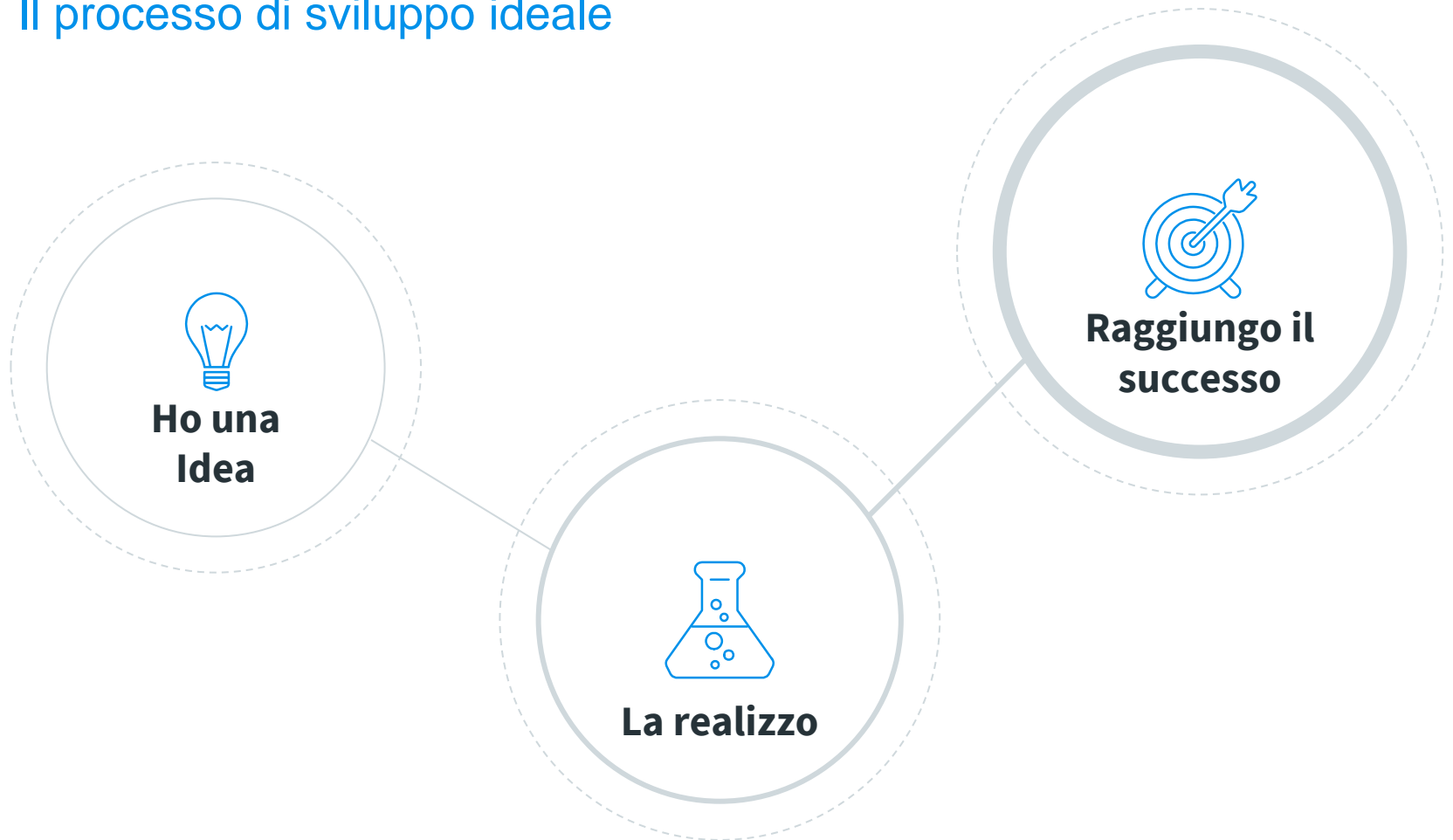
Circa 2.860.000 risultati (0,53 secondi)

Web Programming Is Getting Unnecessarily Complicated | Hacker News
<https://news.ycombinator.com/item?id=12118593>  Traduci questa pagina
18 lug 2016 - I wonder how many are like me (old school? Just old??) and eschew platforms and libraries and roll my own most of the time, because most of the time, the needs are simple, and it's quicker to pound something out rather than learn and use Yet Another Framework. Though using jquery is nice sometimes.

Web programming is getting unnecessarily complicated : programming ...
https://www.reddit.com/.../programming/.../web_programming...  Traduci questa pagina
18 lug 2016 - I think there is a general rule that as development time and resources are a constant. When the web was new someone would spend a large amount of time hand programming a simple website. With new tools we can spend that time making complex sites. Many modern sites would have been desktop ...

Are we making the **web too complicated?** : programming 24 mag 2017
We've Made **Web Development Complicated** : programming 14 mag 2015
For **web developers**, what's are some real-world applications ... 26 nov 2014
[Altri risultati in www.reddit.com](#)

Il processo di sviluppo ideale



Il processo di sviluppo reale (semplificato)

Idea

- Strategia
- Monetizzazione

Analisi

- Requirements
- Team
- Roadmap

Design

- UserExperience
- UserInterface

Mock

- Requirements
- Roadmap
- Realizzazione
- Feedback

Sviluppo

- Metodologia
- Backend
- Frontend
- Amministrazione

Testing

- Validazione requirements
- Beta phase
- Analytics

Deploy

- Cloud
- Store

Supporto

Use Cases (semplificato)



- Per l'utente:
 - Facile da rintracciare
 - Facile da installare
 - Facile da usare
 - Riconoscibile (con una propria identità)
 - Sicura
 - Stabile
 - Veloce
 - Poco energivora
 - Leggera nei trasferimenti
 - Con notifiche
 - Facile da condividere
 - Backup automatico



- Per lo sviluppatore:
 - Facile da mantenere
 - Facile da aggiornare
 - Che sia scalabile al crescere degli utenti
 - Che sia economica (cloud/server)
 - Sicura
 - Stabile
 - Che rispetti le linee guida degli store
 - A/B Test facile da integrare
 - Logger e altri servizi facili da integrare



- Per l'amministratore:
 - Facile da usare
 - Facile da analizzare
 - Facile da mantenere

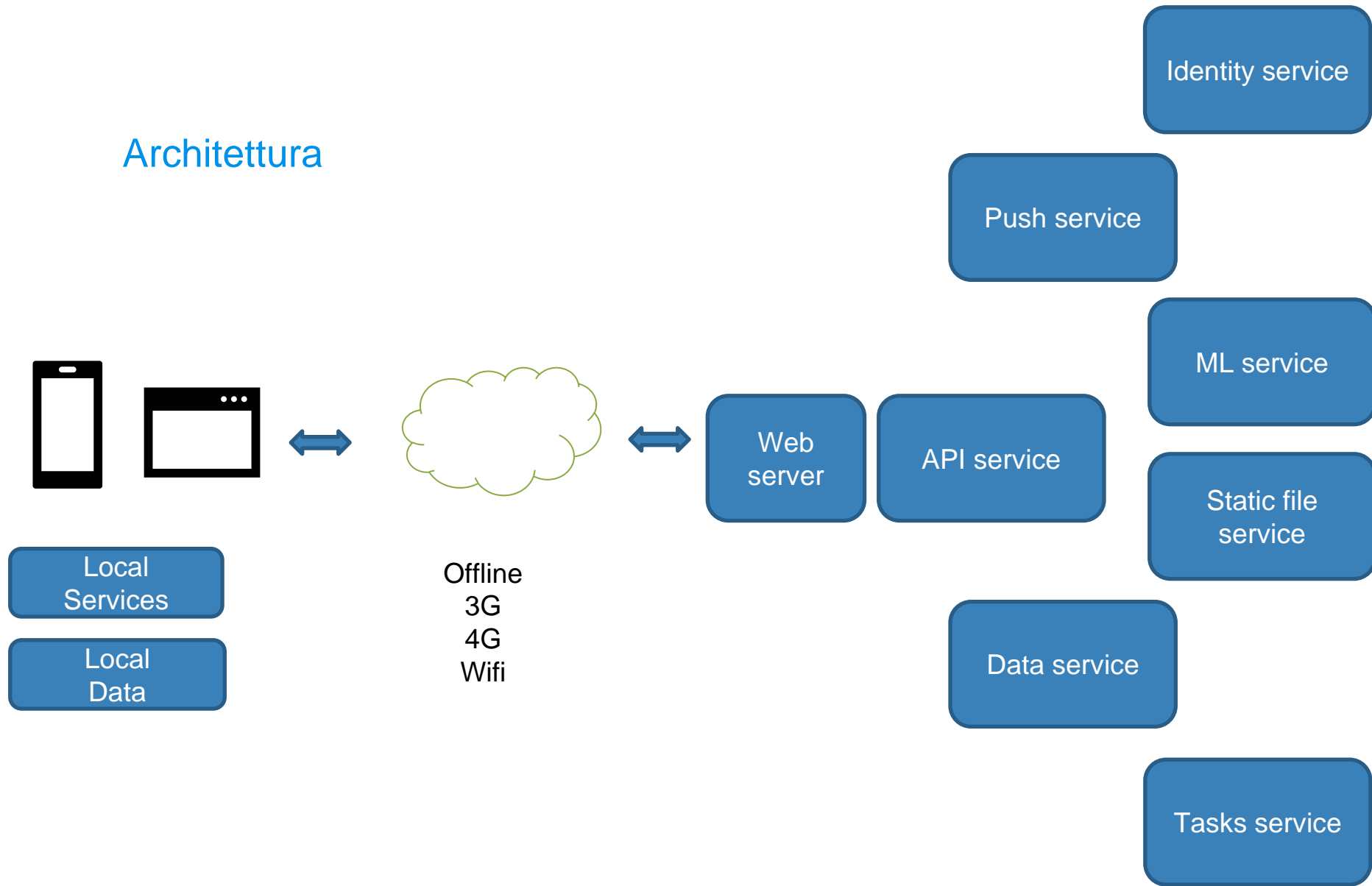


- Per il customer service
 - Facile da usare

A decorative network diagram in the top-left corner, consisting of various sized circles (nodes) connected by thin lines (edges). Some nodes are solid grey, while others are hollow with a grey outline. The connections form a complex, interconnected web.

2. Architettura

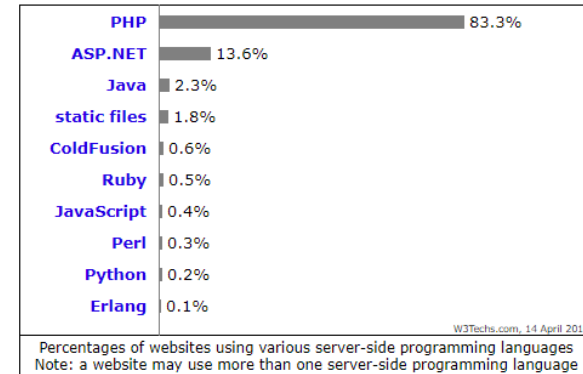
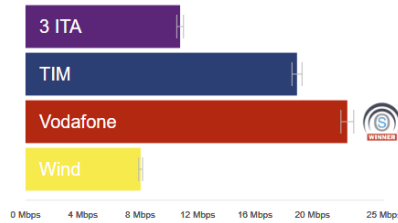
Architettura



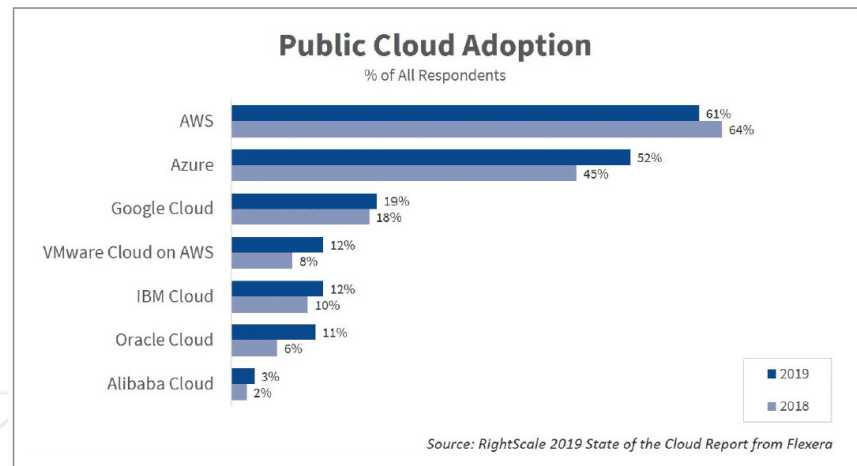
Diffusione tecnologie



Download Speed: Overall OpenSignal



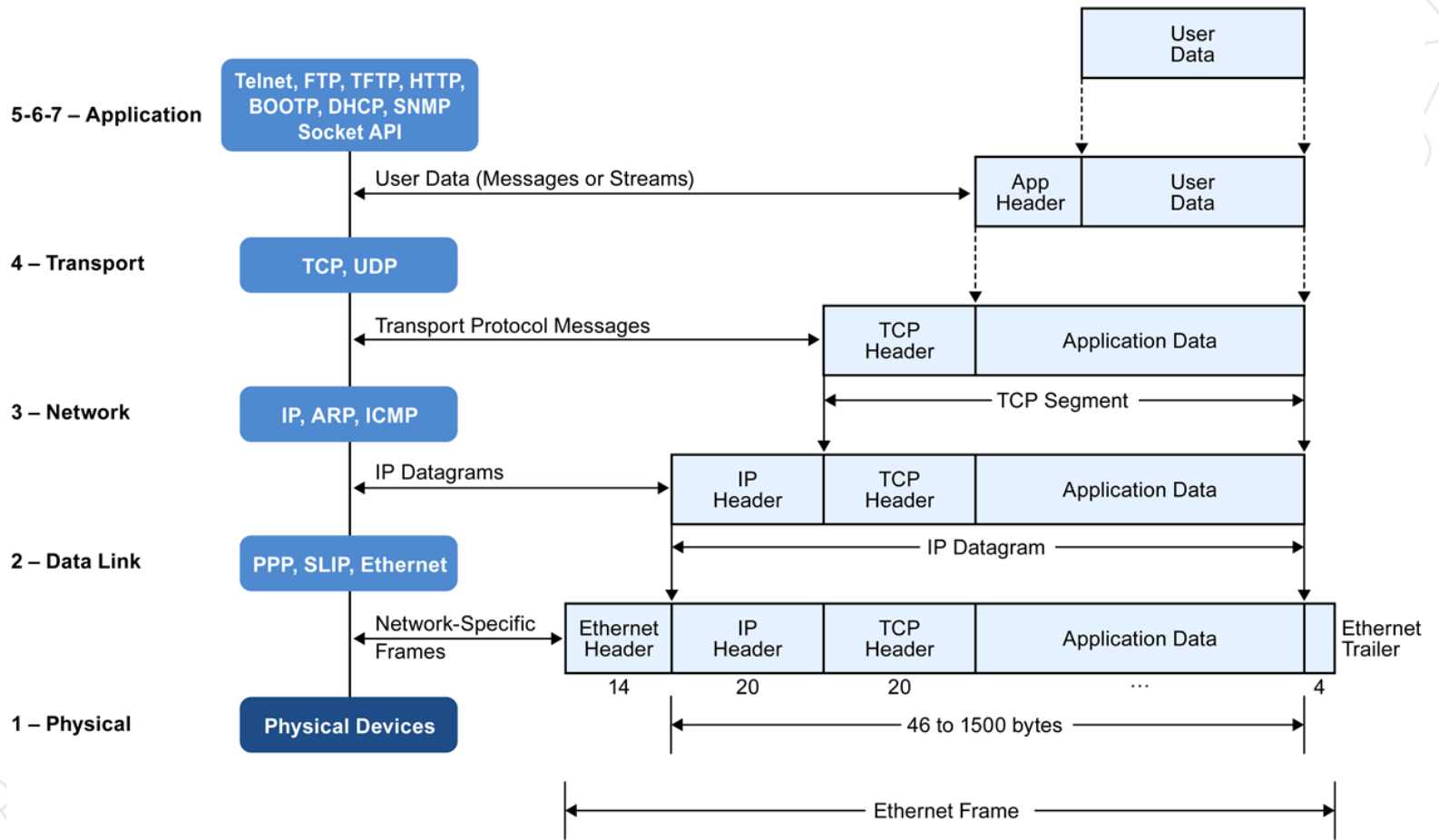
W3Techs.com, 14 April 2018



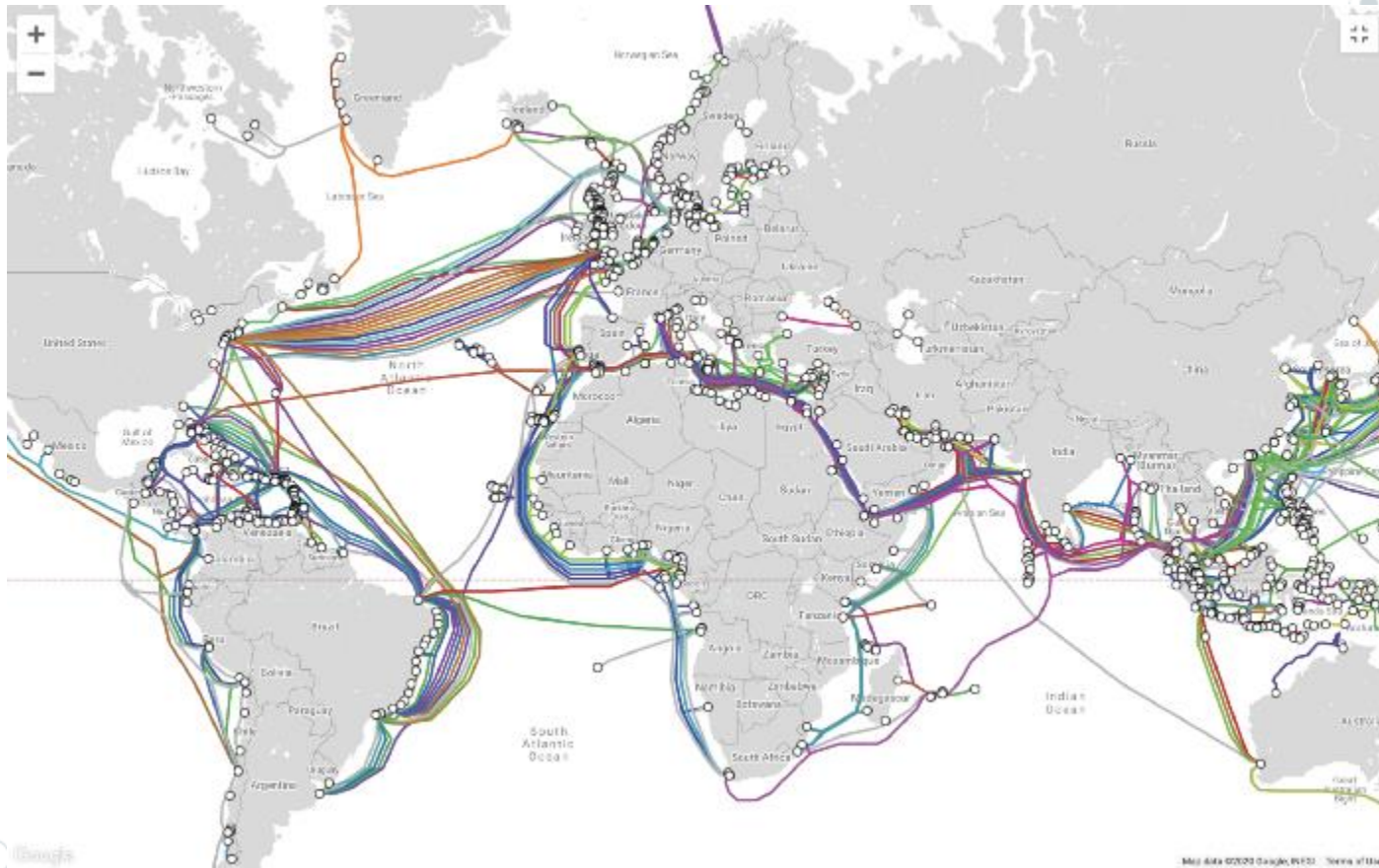
Protocolli



Modello ISO/OSI



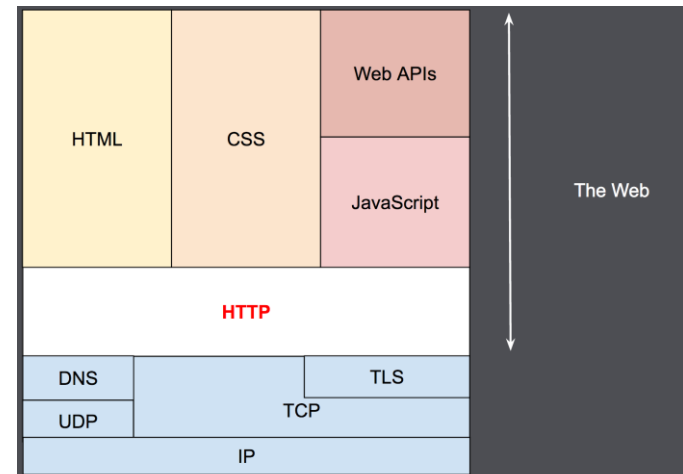
Che giro!



<https://betterprogramming.pub/understand-the-flow-of-a-http-request-1a268ec193f0>

HyperText Transfer Protocol (HTTP - rfc2616)

- Protocollo a livello applicativo
- A livello di trasporto si basa sul TCP (o TLS)
- Request/Response (Client / Server)
- Url composta da http://host:port/path/file
- Metodo: GET/POST/PUT/DELETE/OPTIONS..
- Stato nella risposta: 200/300/400/404/500
- Header di request e di response
- Gestione cookie
- Diversi content-type (html/text/image/json/xml)



HyperText Transfer Protocol

(1) User issues URL from a browser
http://host:port/path/file



(5) Browser formats the response and displays

Client (Browser)

(2) Browser sends a request message

```
GET URL HTTP/1.1  
Host: host:port  
.....  
.....
```

(4) Server returns a response message

```
HTTP/1.1 200 OK  
.....  
.....
```

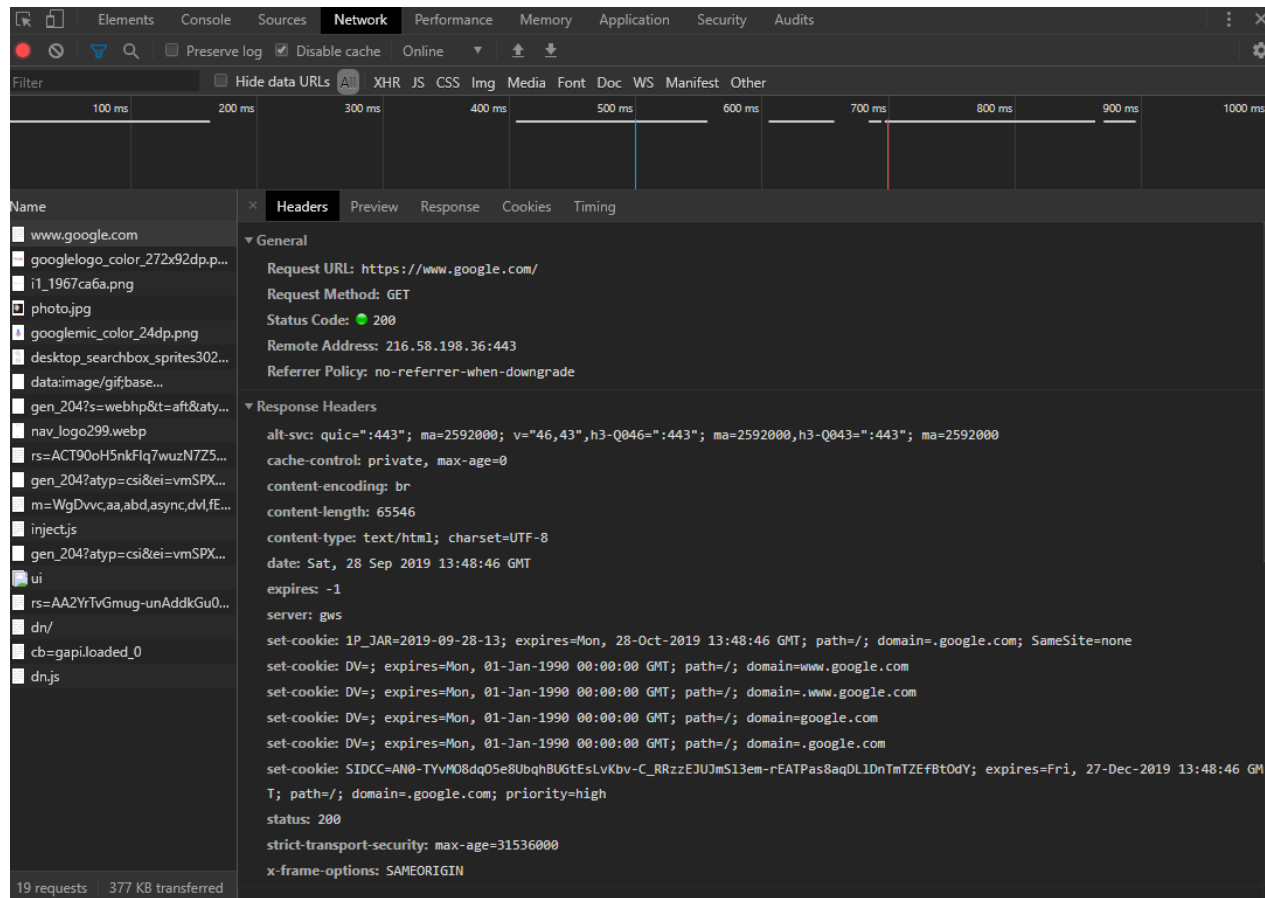
HTTP (Over TCP/IP)

(3) Server maps the *URL* to a file or program under the document directory.

Server (@ *host:port*)

HyperText Transfer Protocol

Studiare:
Headers – Metodi - Cookie – Status Code - Timing



The screenshot shows the Chrome DevTools Network tab with the 'Headers' sub-tab selected. The request is a GET to `https://www.google.com/` with a status code of 200. The response headers are as follows:

```
alt-svc: quic=":443"; ma=2592000; v="46,43",h3-Q046=":443"; ma=2592000,h3-Q043=":443"; ma=2592000
cache-control: private, max-age=0
content-encoding: br
content-length: 65546
content-type: text/html; charset=UTF-8
date: Sat, 28 Sep 2019 13:48:46 GMT
expires: -1
server: gws
set-cookie: 1P_JAR=2019-09-28-13; expires=Mon, 28-Oct-2019 13:48:46 GMT; path=/; domain=.google.com; SameSite=none
set-cookie: DV=; expires=Mon, 01-Jan-1990 00:00:00 GMT; path=/; domain=www.google.com
set-cookie: DV=; expires=Mon, 01-Jan-1990 00:00:00 GMT; path=/; domain=.google.com
set-cookie: DV=; expires=Mon, 01-Jan-1990 00:00:00 GMT; path=/; domain=google.com
set-cookie: DV=; expires=Mon, 01-Jan-1990 00:00:00 GMT; path=/; domain=.google.com
set-cookie: SIDCC=AN0-TYVM08dq05e8UubqhBUGtEslvKbv-C_RRzzEJUJmS13em-rEATPas8aqDL1DnTmTzEFbt0dY; expires=Fri, 27-Dec-2019 13:48:46 GMT; path=/; domain=.google.com; priority=high
status: 200
strict-transport-security: max-age=31536000
x-frame-options: SAMEORIGIN
```

HyperText Transfer Protocol

GET www.google.it

Untitled Request

GET www.google.it

Params Authorization Headers (9) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies (2) Headers (12) Test Results

Status: 200 OK Time: 69ms Size: 5.57 KB Save Response

KEY	VALUE
Date	Sat, 28 Sep 2019 14:00:55 GMT
Expires	-1
Cache-Control	private, max-age=0
Content-Type	text/html; charset=ISO-8859-1
P3P	CP="This is not a P3P policy! See g.co/p3phelp for more info."
Content-Encoding	gzip
Server	gws
Content-Length	4996
X-XSS-Protection	0
X-Frame-Options	SAMEORIGIN
Set-Cookie	1P_JAR=2019-09-28-14; expires=Mon, 28-Oct-2019 14:00:55 GMT; path=/; domain=.google.it; SameSite=none
Set-Cookie	NID=188=bYVhzxIbugZ36jBJEs90gZwulQ8oVHVIVPwDzr4d-JQkMvk-hFF75qXqnZWQjmq-mLeIKe3NoLb9_UN4oNnZMCm6fWI3jTmNbM7...

HyperText Transfer Protocol

Limiti del protocollo:

- Una connessione per request/response
- Mancanza di gestione delle priorità su connessioni multiple
- Bassa compressione (no header compression)

Es: Apache Web Server Settings

Concurrent Connections

By default apache2 is configured to support 150 concurrent connections. This forces all parallel requests beyond that limit to wait. Especially if, for example, active sync clients maintain a permanent connection for push events to arrive.

This is an example configuration to provide 8000 concurrent connections.

```
<IfModule mpm_worker_module>
  ServerLimit          250
  StartServers         10
  MinSpareThreads     75
  MaxSpareThreads     250
  ThreadLimit         64
  ThreadsPerChild     32
  MaxRequestWorkers   8000
  MaxConnectionsPerChild 10000
</IfModule>
```

Browsers:

Version	Maximum connections
Internet Explorer® 7.0	2
Internet Explorer 8.0 and 9.0	6
Internet Explorer 10.0	8
Internet Explorer 11.0	13
Firefox®	6
Chrome™	6
Safari®	6
Opera®	6
iOS®	6
Android™	6

HTTP2 - rfc7540

Multiplexing

Upwork

HTTP 1.1

3 TCP CONNECTIONS



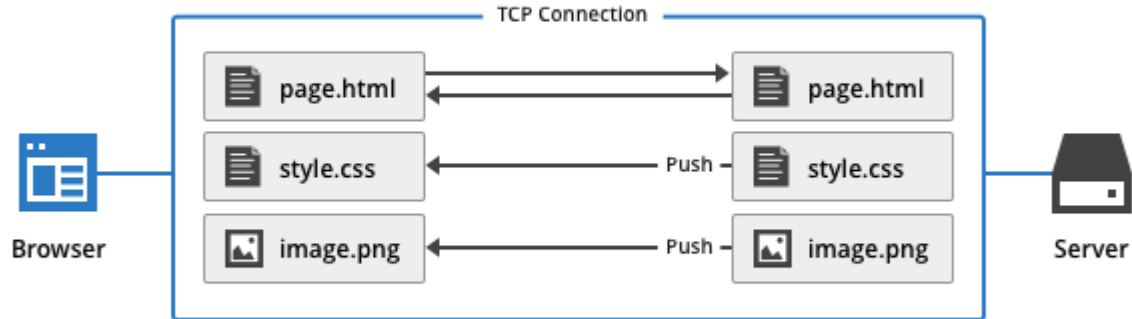
HTTP/2

1 TCP CONNECTION

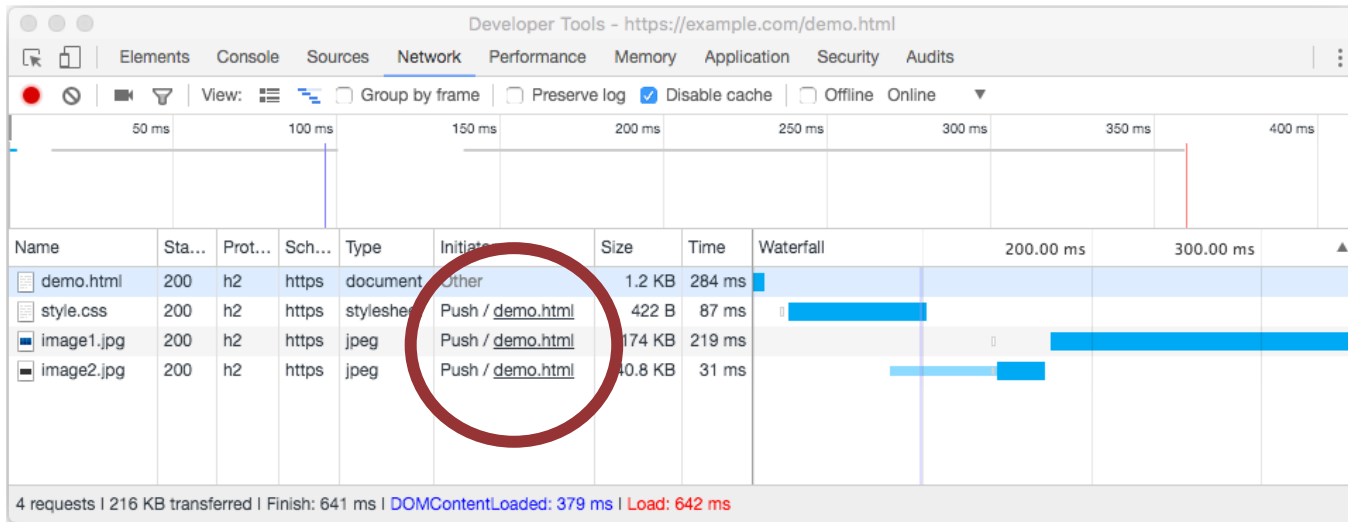


HTTP2

HTTP/2 (With Server Push)

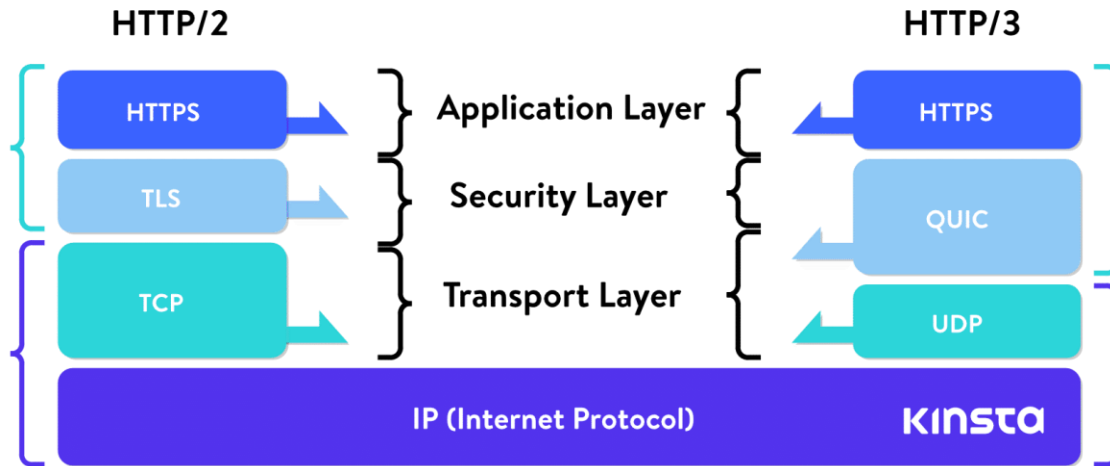


Single TCP Connection, Single HTTP Request

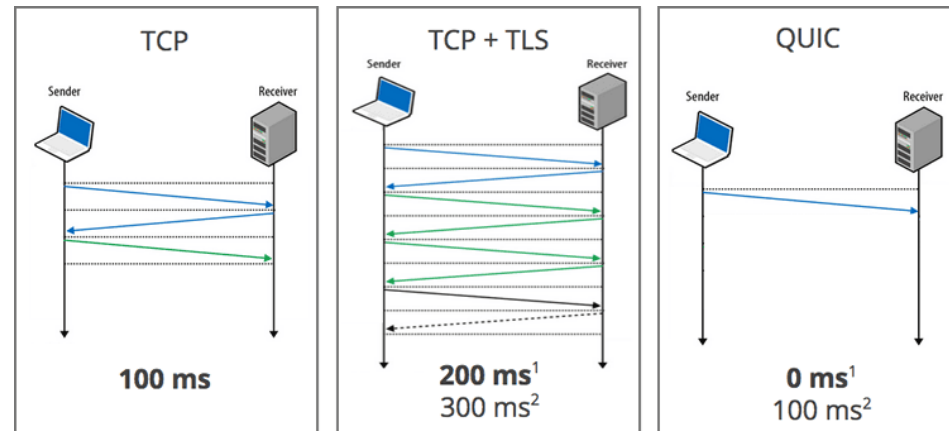


<https://http2.akamai.com/demo/http2-lab.html>

HTTP3



HTTP/3 è la terza versione dell'Hypertext Transfer Protocol (HTTP), già noto come HTTP-over-QUIC. QUIC (Quick UDP Internet Connections) è stato inizialmente sviluppato da Google ed è il successore di HTTP/2. Aziende come Google e Facebook stanno già utilizzando QUIC per velocizzare il web.



<https://kinsta.com/it/blog/http3/>
<https://www.evemilano.com/protocolli-http/>

WebSocket - rfc6455

Limiti del protocollo:

- Primo handshake su http
- Se tutto va bene il protocollo della connessione passa da http a websocket (usando la connessione Tcp precedentemente aperta dalla prima connessione http)
- A questo punto rimane solo il protocollo websocket
- Scambio messaggi bidirezionale

