

Tehnologii Web



servicii Web (III)

micro-servicii • *serverless* • GraphQL

„Toate erau laolaltă – după aceea, a venit mintea
și le-a pus în ordine.”

Anaxagoras

dezvoltarea aplicațiilor Web

Serviciu Web

software – utilizat la distanță de alte aplicații/servicii –
oferind o funcționalitate specifică

implementarea sa nu trebuie cunoscută
de programatorul ce invocă serviciul

a se revedea
prelegerile anterioare

Există alternative la servicii Web?

micro-servicii

Implementează o funcționalitate specifică,
oferită la nivel de unic proces

self-contained system

componentă la nivel de *backend* dezvoltată cu scopul
de a fi **înlocuită**, nu de a fi reutilizată

small

each running in its own process

lightweight communication mechanisms (usual, HTTP)

built around business capabilities

independently deployable

minimum of centralized management

may be written in different programming languages

may use different data storage mechanisms

caracteristici ale micro-serviciilor conform

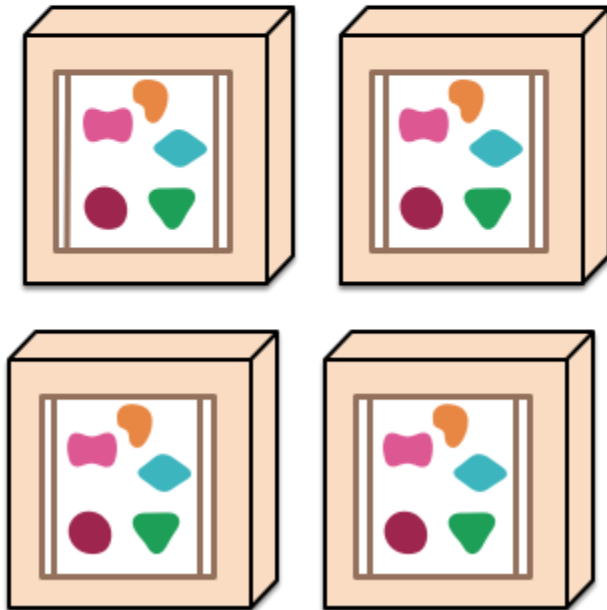
James Lewis & Martin Fowler, *Microservices* (2014)

martinfowler.com/articles/microservices.html

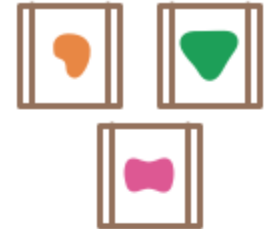
A monolithic application puts all its functionality into a single process...



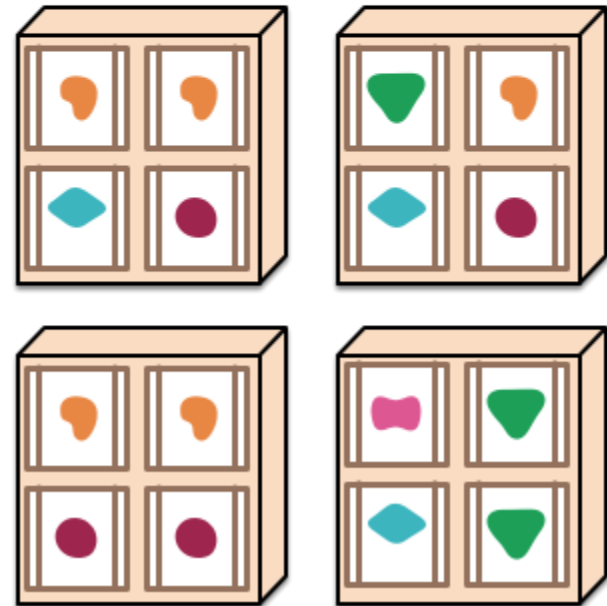
... and scales by replicating the monolith on multiple servers



A microservices architecture puts each element of functionality into a separate service...



... and scales by distributing these services across servers, replicating as needed.



modularitate, descentralizare și evoluție permanentă

exemple de bună practică: microservices.io

Beneficii ale micro-serviciilor:

izolare
autonomie
scalabilitate individuală
reziliență
viteză
suport pentru experimentare
feedback rapid
flexibilitate
ușor de înlocuit
ecosistem

S. Tilkov, *A Question of Size – Modularization & Microservices*,
Java Forum Nord 2017:

speakerdeck.com/stilkov/a-question-of-size-modularization-and-microservices

micro-servicii

Funcționale (*functional services*)

implementează funcționalități specifice
(*business operations*)

micro-servicii

Funcționale (*functional services*)

implementează funcționalități specifice
(*business operations*)

expuse consumatorului de servicii

independente (fără efecte colaterale – *no side effects*)

nu sunt partajabile uzual

micro-servicii

Control – infrastructură (*infrastructure services*)

implementează activități non-funcționale:
autentificare, autorizare, jurnalizare, monitorizare,...

micro-servicii

Control – infrastructură (*infrastructure services*)

implementează activități non-funcționale:
autentificare, autorizare, jurnalizare, monitorizare,...

nu sunt expuse în exterior – private

pot fi partajate la nivel de aplicație ori servicii interne

micro-servicii

Aspect de interes:

partajarea funcționalităților

share-as-much-as possible (SOA clasic)

versus

share-as-little-as possible (micro-servicii)

micro-servicii

Aspect de interes:

comunicarea – uzual, asincronă – între (micro-)servicii

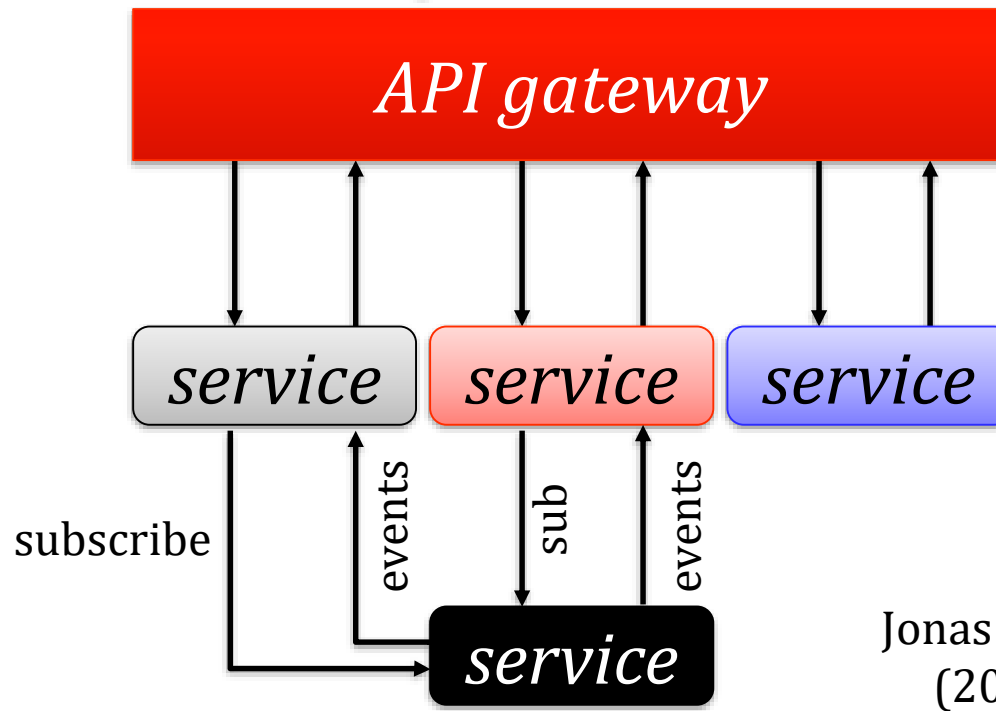
abordări:

point-to-point

sau

publish-subscribe

simplificarea
accesului clientului
via API



Jonas Bonér
(2016)

intern, (micro-)serviciile pot comunica
recurgând la *publish-subscribe*

WebSub (recomandare W3C, 2018): www.w3.org/TR/websub/

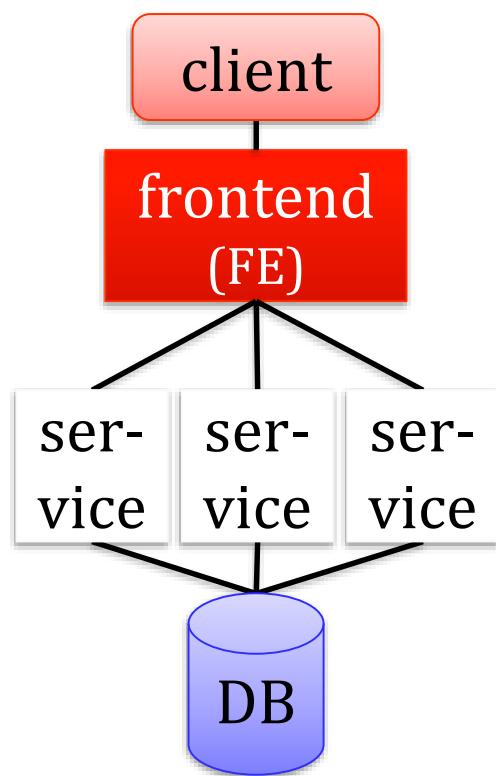
micro-servicii

Uzual, arhitecturile ce recurg la micro-servicii
nu includ componente *middleware*
și nu oferă suport pentru abstractizarea interacțiunii
dintre producătorii și consumatorii de servicii
(*contract decoupling*)

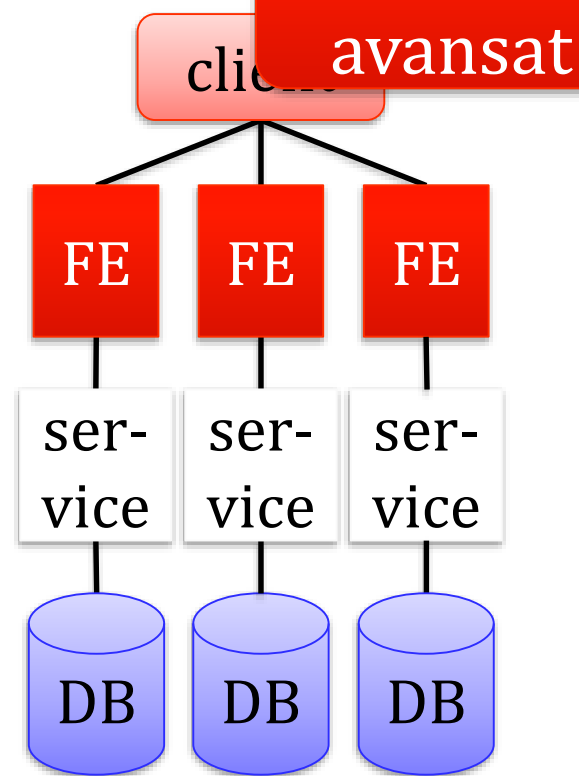
micro-servicii

Uzual, arhitecturile ce recurg la micro-servicii
nu includ componente *middleware*
și nu oferă suport pentru abstractizarea interacțiunii
dintre producătorii și consumatorii de servicii
(*contract decoupling*)

- ▶ **μSOA** – *Microservice Oriented Architecture*



arhitectură bazată
pe servicii Web



arhitectură recurgând
la microservicii

Z. Dehghani, *How to break a Monolith into Microservices* (2018)
martinfowler.com/articles/break-monolith-into-microservices.html

cazuri concrete: Amazon, Groupon, Netflix, Twitter,...
de studiat prezentările lui Stefan Tilkov: speakerdeck.com/stilkov

micro-servicii: dezvoltare

Platforme:

Cocaine, Deis, Fabric8, Hook.io, OpenWhisk,...

Framework-uri:

Akka, Baratine, Finagle, Ice, Orbit, Vert.X, Wangle etc.

SDK-uri multi-limbaj:

Apex, CoAP, gRPC, Hprose

multe alte instrumente software enumerate la

github.com/mfornos/awesome-microservices

API

Accesul la un (micro-)serviciu are loc uzual pe baza unei interfețe de programare a aplicației
API (*Application Programming Interface*)

API

Accesul la un (micro-)serviciu are loc uzual pe baza unei interfețe de programare a aplicației
API (***Application Programming Interface***)

“any well-defined interface that defines the service that one component, module, or application provides to other software elements”
(de Souza et al., 2004)

API

API „de succes” – adaptare după (Bloch, 2005)

ușor de învățat

facil de folosit, chiar și în lipsa documentației

previne utilizarea eronată

stabil și sigur

ușor de menținut

suficient de expresiv

facil de extins

vezi și S. Clarke, “*Measuring API Usability*”: drdobbs.com/windows/184405654

API

Componentă software concepută și invocată
via tehnologiile Web actuale
(URI, HTTP, formate de date: JSON, XML)

poate fi dezvoltată conform unui stil arhitectural
e.g., REST (*REpresentational State Transfer*)



vezi cursul trecut

implementare

De la aplicații la API-uri și servere de aplicații



Brian Mulloy, *Web API Design*, Apigee, 2016

docs-apis.apigee.io/files/Web-design-the-missing-link-ebook-2016-11.pdf

API

API public

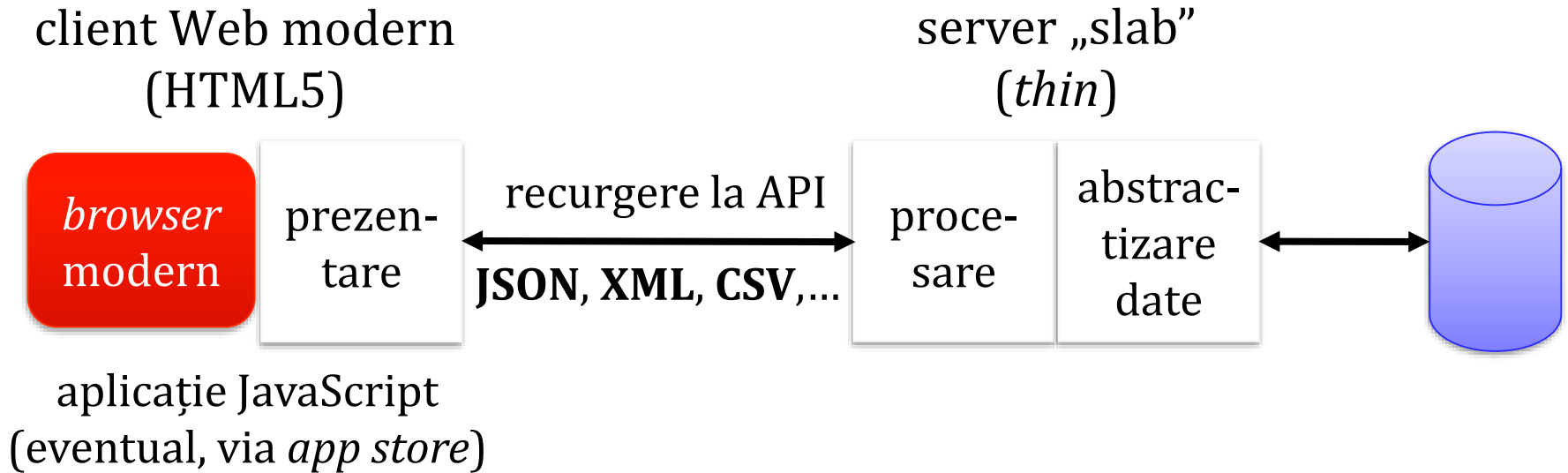
(disponibil pe baza unei licențe de utilizare)

versus

API privat

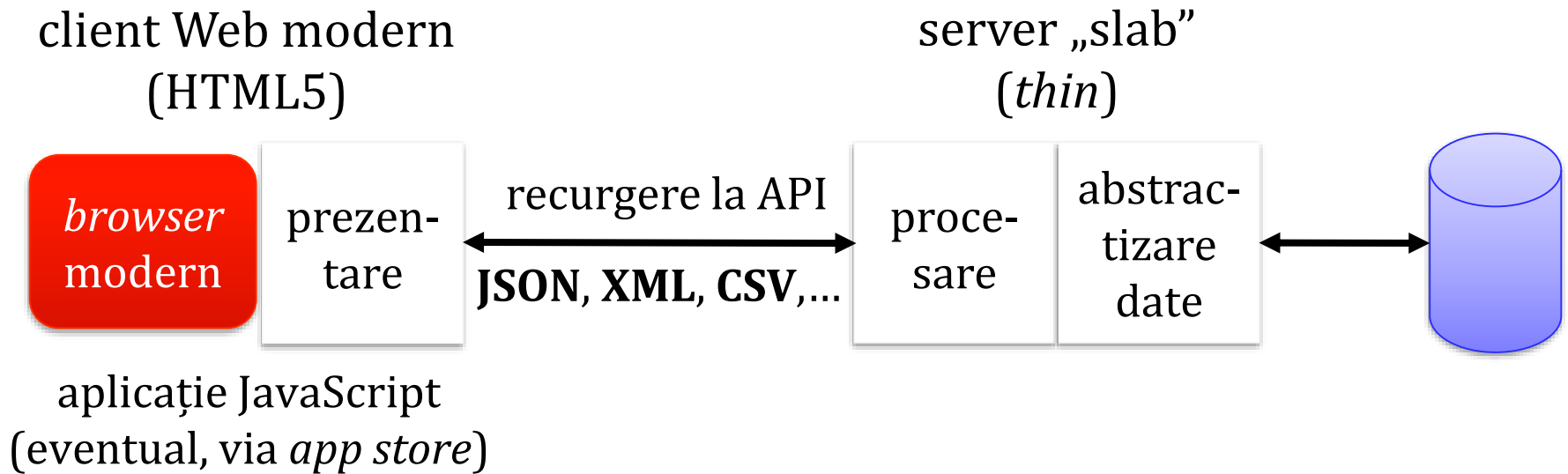
(pentru uz intern)

API: abordare client – JavaScript



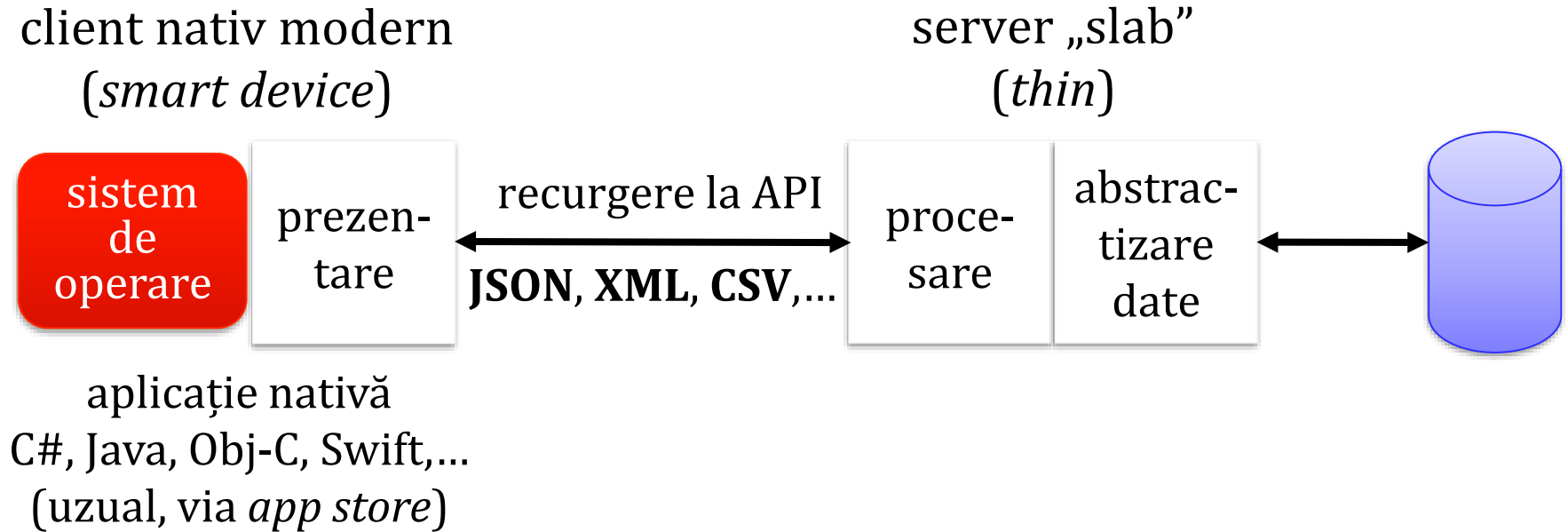
browser Web pe calculatoare
convenționale, dispozitive
mobile și altele

API: abordare client – JavaScript



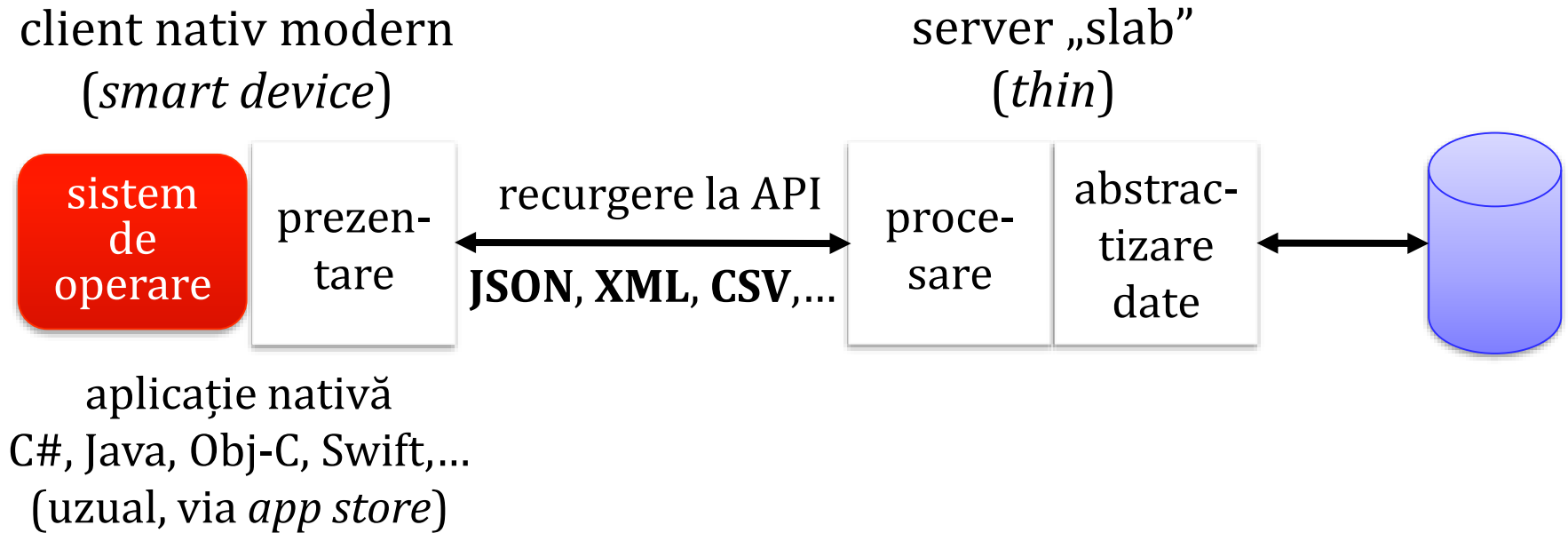
implementarea aplicației JavaScript poate recurge la
biblioteci, *framework*-uri, componente specifice
e.g., Angular, React, Vue

API: aplicații native



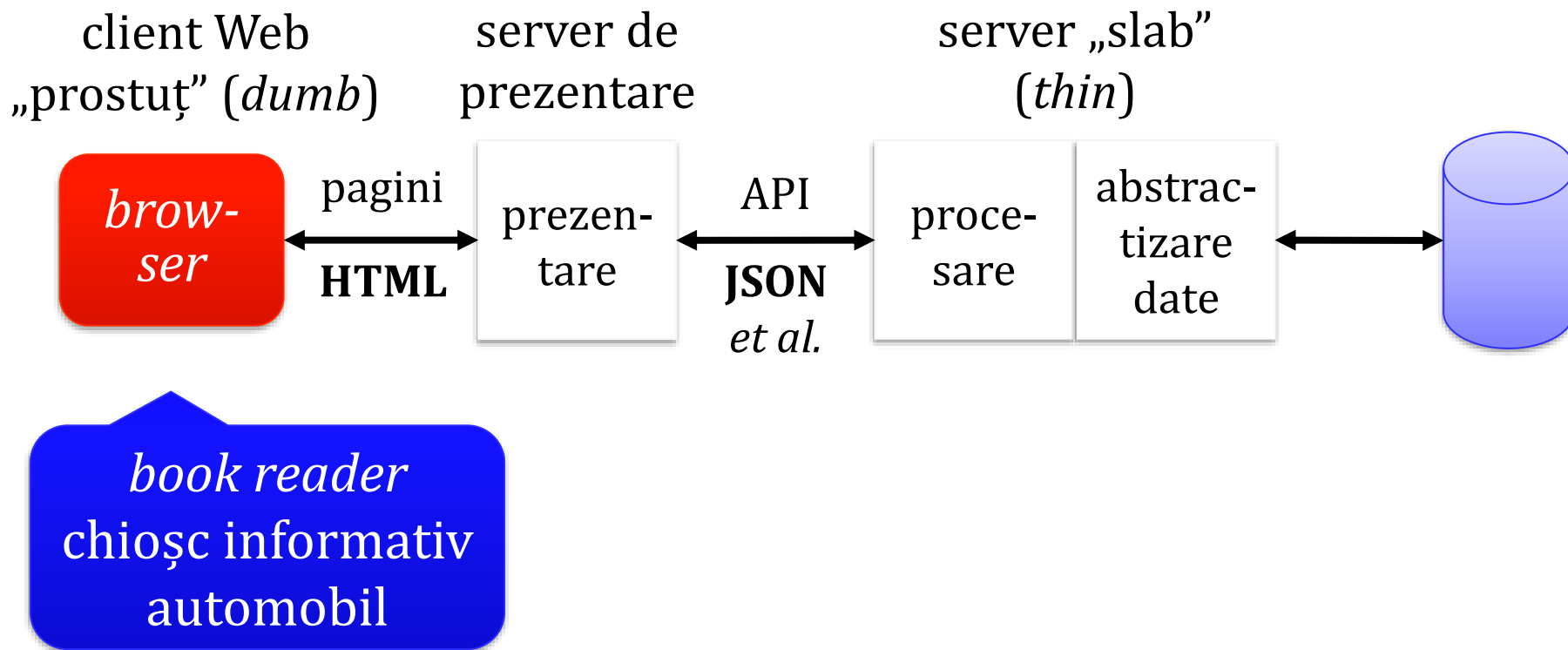
*desktop și/sau mobile,
smart TV, home appliance,
dispozitiv ambiental*

API: aplicații native



implementarea aplicației native poate recurge la biblioteci, *framework*-uri, componente specifice *e.g.*, Apache Cordova, Flutter, Ionic, React Native, NW.js

API: abordare bazată pe intermediari



API: în contextul *serverless*

Aplicația depinde semnificativ de componente externe, disponibile în „nori”

(micro-)servicii expuse via API



abordarea *serverless*

API: în contextul *serverless*

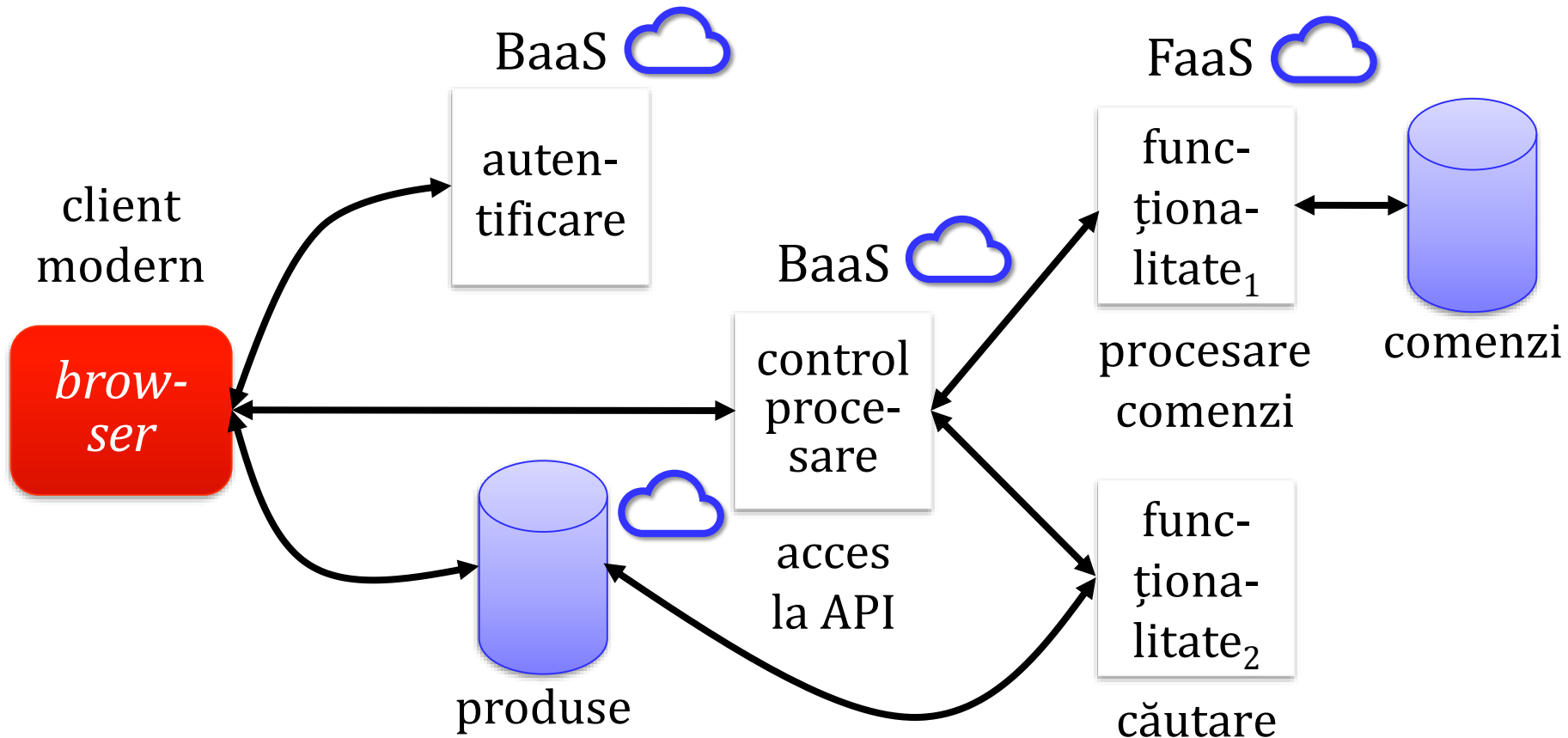
Serverless

strat de abstractizare a accesului la resursele
unei platforme de tip *cloud*

Mike Roberts (2018)

martinfowler.com/articles/serverless.html

API: în contextul *serverless*



BaaS = *(Mobile) Backend As A Service* **FaaS** = *Functions As A Service*

API: în contextul *serverless*

FaaS – *Functions As A Service*

funcții (*cloud functions*) implementând funcționalități
expuse consumatorului de servicii

as small as possible

uzual, implementări sub 100 de linii de cod

API: în contextul *serverless*

FaaS – *Functions As A Service*

executate – la nivel de server – independent și asincron,
fără a cauza efecte colaterale

declanșate de evenimente

utilizatorul nu e preocupat de managementul resurselor
și alte sarcini

API: în contextul *serverless*

BaaS – *Backend As A Service*

încapsulează servicii de infrastructură ce implementează
activități non-funcționale
(autentificare, autorizare, jurnalizare, monitorizare etc.)

private – nu sunt expuse în exterior

pot fi partajate de serviciile interne

API: în contextul *serverless*

Serverless computing = FaaS + BaaS

a se consulta și articolul

Sabin Buraga, *Aspecte arhitecturale*

vizând dezvoltarea de aplicații serverless (2019)

ittransfer.space/aspecte-arhitecturale-vizand-dezvoltarea-de-aplicatii-serverless/

resurse + soluții software:

github.com/anaibol/awesome-serverless

Cum poate fi descrisă interfața unui API?

API: descriere abstractă

OpenAPI Specification (ex-Swagger) – openapis.org

RAML (*RESTful API Modeling Language*) – raml.org

API Blueprint – apiblueprint.org

alte resurse de interes:

github.com/Kikobeats/awesome-api

API: descriere abstractă

OpenAPI Specification

soluție modernă de a declara – independent de platformă –
interfața publică a unui API REST

versiunea curentă: OpenAPI 3.0 (octombrie 2018)

formate folosite: JSON și/sau YAML

API: descriere abstractă

OpenAPI Specification

biblioteci de procesare – exemple:

KaiZen OpenAPI Parser (Java)

Microsoft.OpenApi.net (C#)

Open API Definition Parser (Ruby)

Spectral (JavaScript, TypeScript)

API: descriere abstractă

OpenAPI Specification

creare de servicii (puncte terminale – *end-points*)
pe baza unui document OpenAPI:

Exegesis (Node.js)

FastAPI (Python)

Fusio (PHP, JavaScript)

PHP-CRUD-API (PHP)

Vert.x (Java, Kotlin, JS, Ruby, Scala,...)

API: descriere abstractă

OpenAPI Specification

generatoare de cod – exemplificări:

BaucisJS (Node.js)

gnostic (Go)

WebSphere Liberty (Java)

ZRO (Ruby on Rails)

```
{
  "openapi": "3.0.0",
  "paths": {
    "/resource": {
      "get": {
        "operationId": "service",
        "parameters": [ {
          "name": "parameter",
          "in": "query",
          "schema": { "type": "string" }
        } ],
        "responses": {
          "200": {
            "description": "Success",
            "schema": {
              "$ref": "#/definitions/Response"
            }
          }
        }
      }
    }
  }
}
```

„scheletul” unui document
OpenAPI specificând un API

```
665 post:
666   summary: Logs in an user
667   description: The Login endpoint allows an user to log in
668   tags:
669     - Login
670   responses:
671     '200':
672       description: The id of the logged user
673       schema:
674         $ref: '#/definitions/LoginResponse'
675     '400':
676       description: >-
677         Bad request. The request could not be understood
678         by the server.
679         Please make a good request.
680       schema:
681         $ref: '#/definitions/ErrorMessage'
682     '403':
683       description: Forbidden. You are forbidden to access
684       this resource.
685       schema:
686         $ref: '#/definitions/ErrorMessage'
687     '405':
688       description: >-
689         Method not allowed. The method specified is not
690         allowed for this
691         resource.
692       schema:
693         $ref: '#/definitions/ErrorMessage'
694     '500':
695       description: >-
696         Internal Server Error. Something bad happened.
697         Please contact the
698         support team.
699       schema:
700         $ref: '#/definitions/ErrorMessage'
701     '501':
702       description: >-
703         Not Implemented. The server does not support
704         this
705         functionality
706         required. Visit us in a few hours, we are
707         preparing a big surprise.
708       schema:
709         $ref: '#/definitions/ErrorMessage'
710     '503':
```

DELETE

/userProfile/{user_id}/enemies/{enemy_id}

Remove
an
enemy

GET

/userProfile/{user_id}/preferences/movies

User movie
preferences

PUT

/userProfile/{user_id}/preferences/movies/{movieType_id}

Add a new
preference

The User preferences endpoint which adds a user preference

Parameters

Try it out

Name

Description

user_id * required

User id.

string
(path)

accesstoken * required

User token.

number
(query)

editarea unei specificații de API
proiectul UReR (V. Vîrlan *et al.*, 2017)

github.com/VirlanValentin/WADe_URer

Responses

Response content type

application/json

```
297: '/live_streams/{id}':
298:   get: {}
273:   patch: {}
356:   delete: {}
414: '/live_streams/{id}/start':
415:   put: {}
499: '/live_streams/{id}/stop':
500:   put: {}
584: '/live_streams/{id}/reset': {}
669: '/live_streams/{id}/regenerate_connection_code': {}
772: '/live_streams/{id}/thumbnail_url': {}
845: '/live_streams/{id}/state': {}
928: '/live_streams/{id}/stats': {}
994: /players: {}
1040: '/players/{id}':
1041:   patch: {}
1042:   summary: Update a player
1043:   description: This operation updates a player.
1044:   operationId: updatePlayer
1045:   tags: {}
1047:   x-code-samples:
1048:     - lang: Shell {}
1050:     - lang: JavaScript {}
1082:   parameters:
1083:     - name: id
1084:       in: path
1085:       type: string
1086:       required: true
1087:       description: The unique alphanumeric string that identifies
         the player.
1088:     - name: player
1089:       in: body
1090:       required: true
1091:       description: Provide the details of the player to update in
         the body of the request.
1092:       schema:
1093:         $ref: '#/definitions/player_update_input'
1094:   responses:
1095:     '200': {}
1104:     '401': {}
1108:     '403': {}
1112:     '404': {}
1116:     '410': {}
1120:     '422': {}
1124:   get: {}
1189: '/players/{id}/rebuild': {}
1269: '/players/{id}/state': {}
1347: '/players/{player_id}/urls': {}
1479: '/players/{player_id}/urls/{id}': {}
```

/players/{id} x

PATCH /players/{id} x

players

Summary

Update a player

Description

This operation updates a player.

Parameters

Name	Located in	Description	Required	Schema
x id	path	The unique alphanumeric string that identifies the player.	Yes	= string
x player	body	Provide the details of the player to update in the body of the request.	Yes	= ▼ player_update_input { player: ▶player { } }

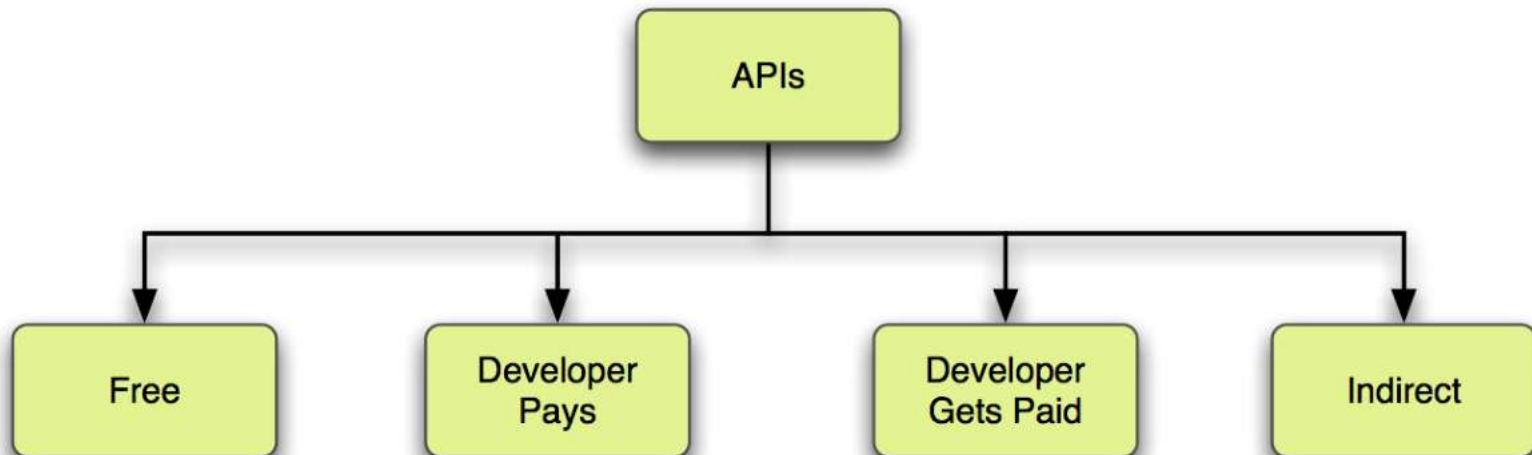
Responses

Code	Description	Schema
x 200	Success	= ▼ { player: ▶player { } }
x 401	Unauthorized	= ▼ Error401 { meta: ▶meta { } }
x 403	Forbidden	= ▼ Error403 { meta: ▶meta { } }

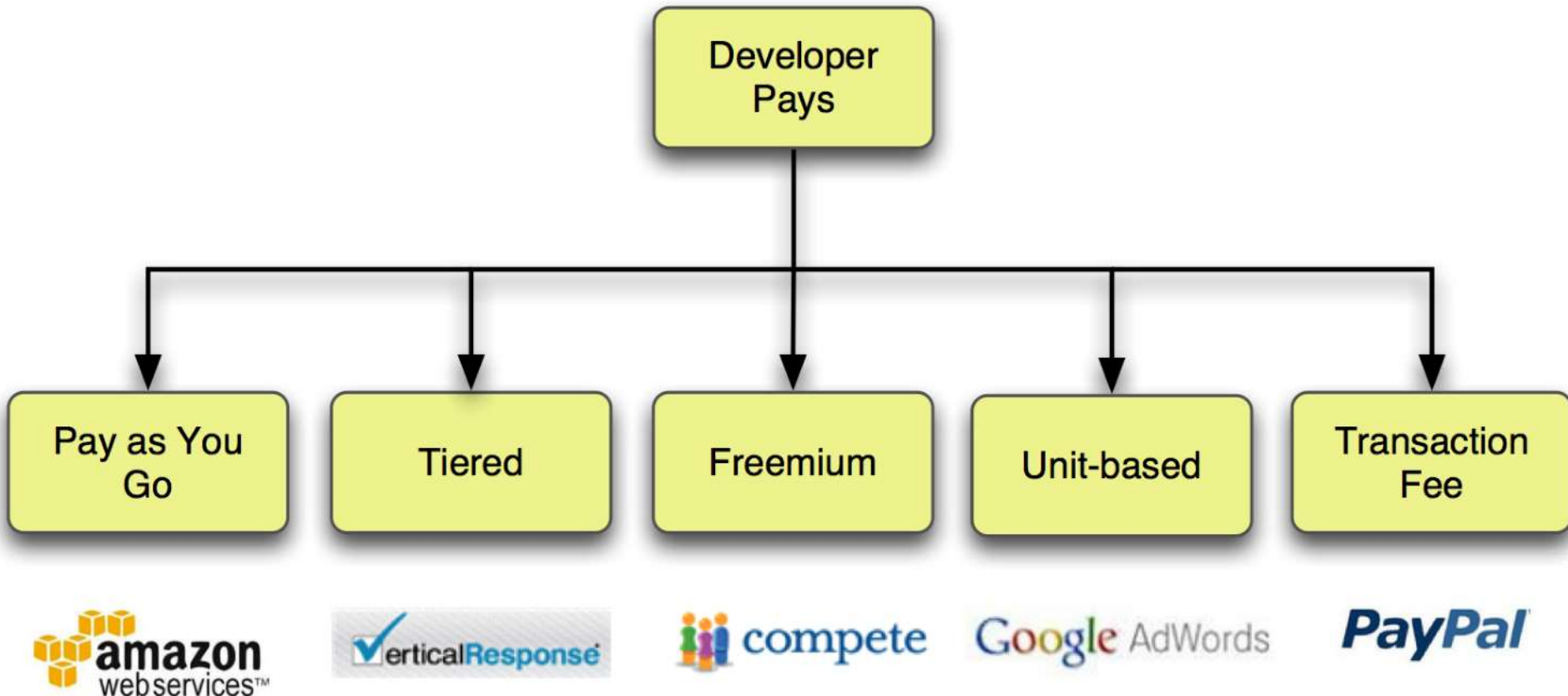
caz concret: **Wowza Streaming Engine REST API**
specificația **OpenAPI** editată cu **{API Studio}**: apistudio.io

implementare

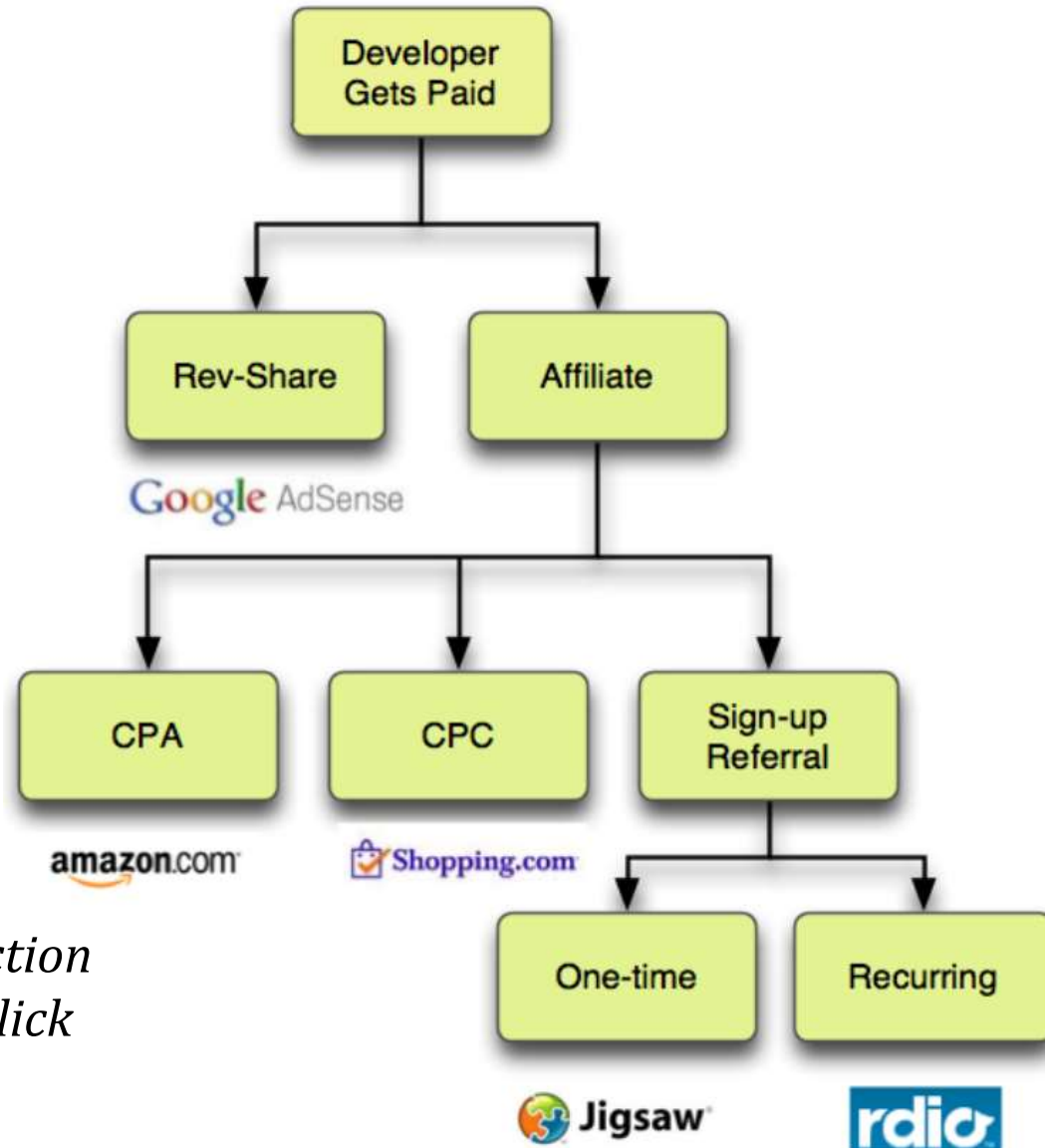
Modele tradiționale de afaceri vizând API-urile



Modele actuale de afaceri privind API-urile



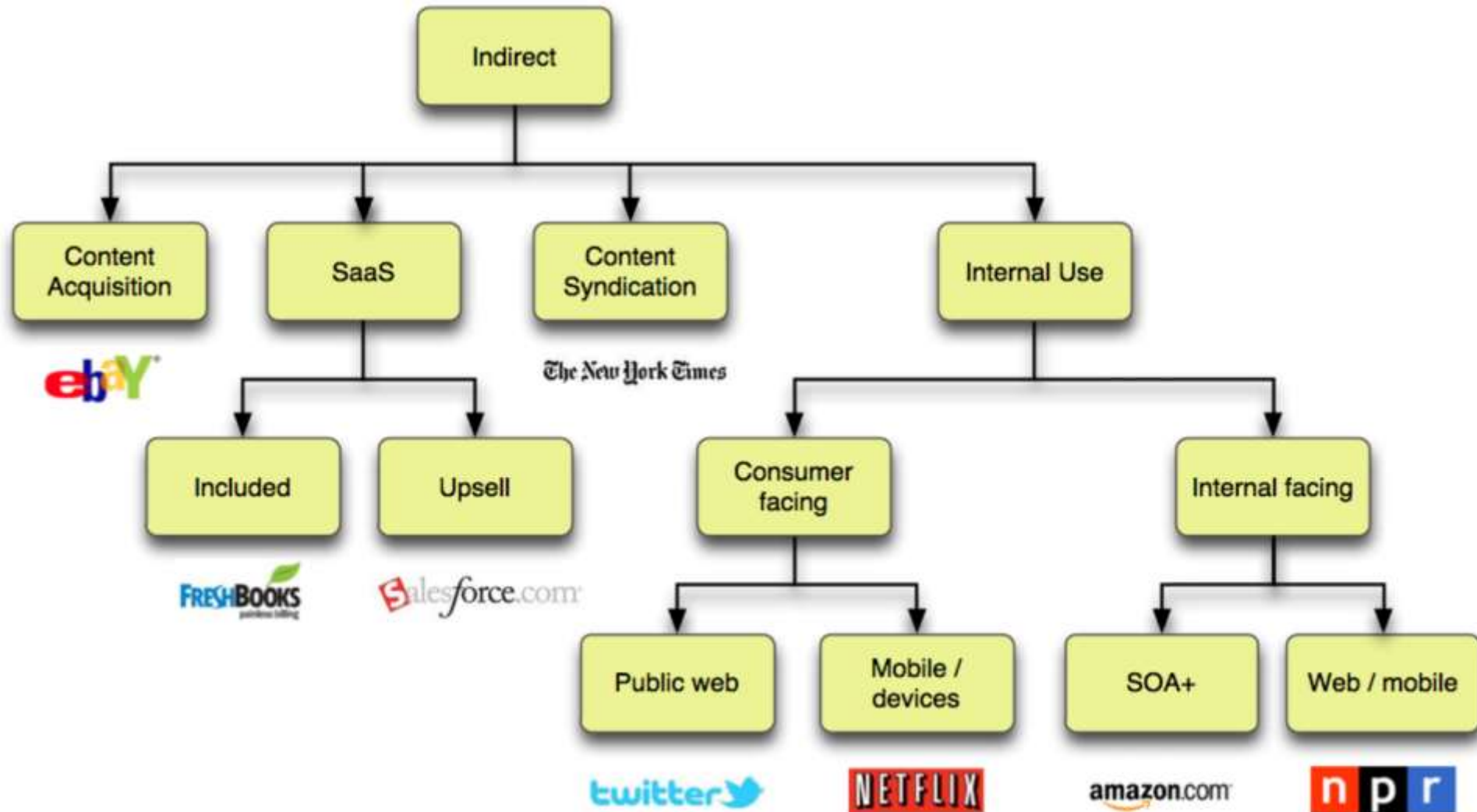
Modele actuale de afaceri privind API-urile



CPA – *cost per action*

CPC – *cost per click*

Modele actuale de afaceri privind API-urile



Putea utiliza servicii Web (API-uri)
pentru autorizare și autentificare?

autorizare

Etape esențiale:

obținere cheie de acces 



autentificarea + autorizarea aplicației  



obținerea acordului utilizatorului   



apelarea funcționalităților serviciului (via API)

autorizare

Pași uzuali de urmat pentru implementarea unei aplicații ce va invoca un serviciu Web pe baza unui API public:

(1) înregistrarea aplicației concepute
via situl entității furnizoare a serviciului

autorizare

Pași uzuali de urmat pentru implementarea unei aplicații ce va invoca un serviciu Web pe baza unui API public:

(1) înregistrarea aplicației concepute
via situl entității furnizoare a serviciului

► cheie de acces – *API key, consumer key, developer key*

autorizare

Pași uzuali de urmat pentru implementarea unei aplicații ce va invoca un serviciu Web pe baza unui API public:

(2) pe baza acestei chei, aplicația se va putea autentifica pentru a putea fi autorizată să acceseze serviciul dorit

autorizare

Pași uzuali de urmat pentru implementarea unei aplicații ce va invoca un serviciu Web pe baza unui API public:

(2) pe baza acestei chei, aplicația se va putea autentifica pentru a putea fi autorizată să acceseze serviciul dorit

pot fi impuse diverse politici de acces (*permissions*): doar consultare (*read*), posibilitatea editării etc.

autorizare

Pași uzuali de urmat pentru implementarea unei aplicații ce va invoca un serviciu Web pe baza unui API public:

(3) autentificarea și autorizarea aplicației
au loc cu acordul utilizatorului

autorizare

Pași uzuali de urmat pentru implementarea unei aplicații ce va invoca un serviciu Web pe baza unui API public:

(3) autentificarea și autorizarea aplicației
au loc cu acordul utilizatorului

dacă utilizatorul nu este autentificat, i se vor solicita informațiile de autentificare (*e.g.*, nume + parola)
– eventual, folosind **2FA** (*Two Factor Auth*) –,
apoi va putea autoriza aplicația să aibă acces la date
via serviciul Web furnizat

autorizare

Pași uzuali de urmat pentru implementarea unei aplicații ce va invoca un serviciu Web pe baza unui API public:

- (4) aplicația apelează funcționalitățile oferite de serviciu pentru preluarea/modificarea datelor de interes, conform politicilor de acces

autorizare

Pași uzuali de urmat pentru implementarea unei aplicații ce va invoca un serviciu Web pe baza unui API public:

- (4) aplicația apelează funcționalitățile oferite de serviciu pentru preluarea/modificarea datelor de interes, conform politicilor de acces

sesiunea curentă va fi stabilită și menținută pe baza unor **informații de autentificare** (*auth tokens*)

oauth

Autorizarea unei aplicații să acceseze date private într-un mod standardizat – pe baza tehnologiilor Web actuale – se poate realiza via **OAuth**

oauth

Autorizarea unei aplicații să acceseze date private
într-un mod standardizat – pe baza tehnologiilor
Web actuale – se poate realiza via **OAuth**

protocol deschis – RFC 6749
OAuth 1.0 (2010), OAuth 2.0 (2012)

oauth.net/2/

L. Spyna, *An OAuth 2.0 Introduction for Beginners* (2018):
itnext.io/an-oauth-2-0-introduction-for-beginners-6e386b19f7a9

avansat



Authorize jsbin



jsbin by [jsbin](#)

wants to access your **busaco** account



Personal user data

Email addresses (read-only)



Gists

Read and write access



Authorize jsbin

Authorizing will redirect to
<http://jsbin.com>



Not owned or
operated by GitHub



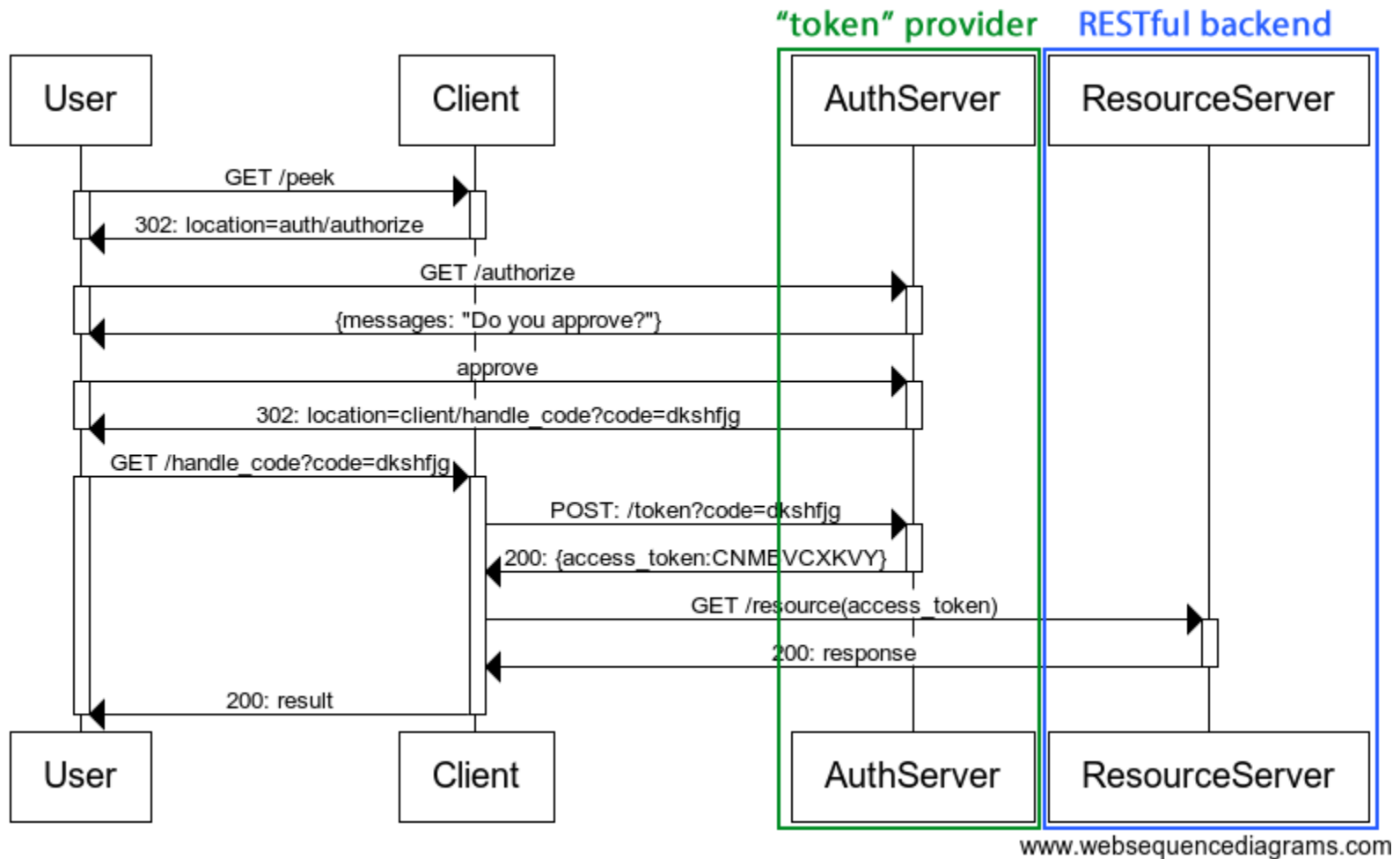
Created
7 years ago



More than 1K
GitHub users

[Learn more about OAuth](#)

autorizarea prin OAuth
a aplicației JS Bin
pentru a obține accesul
la datele unui utilizator
deținute de GitHub



procesele de autorizare a accesului la o resursă Web
conform Dominik Mengelt (2013)



exemplu concret – Facebook:

autorizare cu diverse permisiuni – *e.g.*, age_range, email (acces la adresa de *e-mail* a unui utilizator), public_profile, user_birthday, user_hometown, user_friends, user_likes, user_photos, rsvp_event și altele

developers.facebook.com/docs/facebook-login/permissions/v2.0

oauth

Biblioteci (server și/sau client) disponibile pentru
C, C++, Go, Erlang, Java, JavaScript, Objective-C, .NET,
Perl, PHP, Python, Ruby, Swift,...

oauth.net/code/

oauth









































































Servicii *proxy* de autentificare/autorizare prin OAuth

Auth0 – auth0.com

Hydra – github.com/ory/hydra

OAuth – oauth.io

Okta – developer.okta.com

method	scopes	             
GET me	<input type="checkbox"/> email	             
Get my profile		
GET me/friends	<input type="checkbox"/> friends	             
List my network friends		
GET me/contacts	<input type="checkbox"/> friends	 
List my contacts		
GET me/followers	<input type="checkbox"/> friends	          
List my followers		
GET me/following	<input type="checkbox"/> friends	           
List who i'm following		
GET me/share	<input type="checkbox"/> share	   
List my recent status messages		
POST me/share	<input type="checkbox"/> publish	
Post a status message for me		
POST me/share	<input type="checkbox"/> publish	

exemplificare: **hello.js** – soluție modulară,
la nivel de client, vizând autentificarea și
accesarea serviciilor Web via REST

adodson.com/hello.js/

oauth

Autorizare via servicii Web specifice – exemple:

GitHub – developer.github.com/v3/oauth/

Google – developers.google.com/identity/protocols/OAuth2

LinkedIn – developer.linkedin.com/docs/oauth2

Live Connect (Microsoft) – <http://tinyurl.com/zztr97h>











Stack Exchange – api.stackexchange.com/docs/authentication

Twitter – dev.twitter.com/oauth

WordPress – developer.wordpress.com/docs/oauth2/

Step 1 Select & authorize APIs

Select the scope for the APIs you would like to access or input your own OAuth scopes below. Then click the "Authorize APIs" button.

- ▶  Compute Engine API v1
- ▶  Contacts v3
 - ✓ <https://www.google.com/m8/feeds/>
- ▶  Content API for Shopping v2
- ▶  DCM/DFA Reporting And Trafficking API v3.0
- ▶  Dataflow API v1b3
- ▶  Dialogflow API v2
- ▶  DoubleClick Bid Manager API v1
- ▶  DoubleClick Search API v2
- ▶  Drive API v3
- ▶  Enterprise Apps Reseller API v1

Authorize APIs

Step 2 Exchange authorization code for tokens

Step 3 Configure request to API

Request / Response

```
POST /oauth2/v4/token HTTP/1.1
Host: www.googleapis.com
Content-length: 277
content-type: application/x-www-form-urlencoded
user-agent: google-oauth-playground
```

```
code=4%2FAAC_0qgxCK-LzSISLsmkQ_FSeFi_Cv5iN-cBAgaDlC0WVXtDjaCognBJuaIVxpl
```

```
HTTP/1.1 200 OK
Content-length: 266
X-xss-protection: 1; mode=block
X-content-type-options: nosniff
Transfer-encoding: chunked
Expires: Mon, 01 Jan 1990 00:00:00 GMT
Vary: Origin, X-Origin
Server: GSE
-content-encoding: gzip
Pragma: no-cache
Cache-control: no-cache, no-store, max-age=0, must-revalidate
Date: Fri, 11 May 2018 06:18:22 GMT
X-frame-options: SAMEORIGIN
Alt-svc: hq=":443"; ma=2592000; quic=51303432; quic=51303432; quic=51303
Content-type: application/json; charset=UTF-8
```

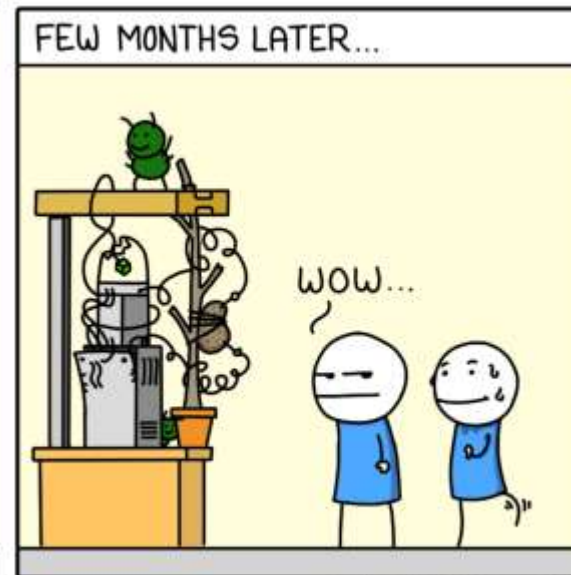
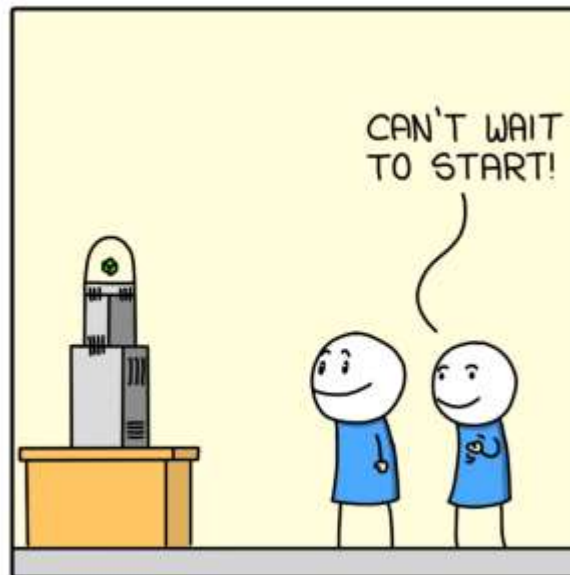
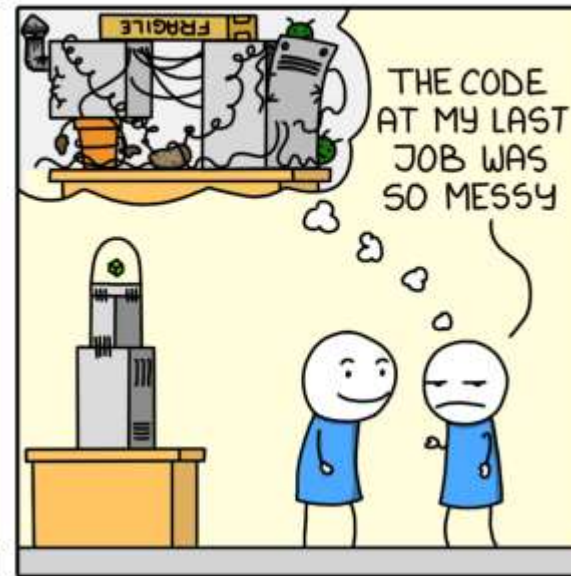
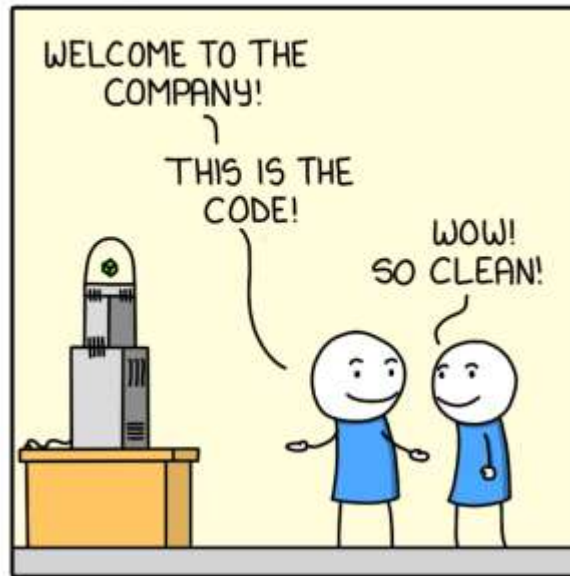
```
{
  "access_token": "ya29.Glu4BQYrldUuKYaEsFifyfhmXTgfvvyBrRlc6jhKN4i6taa6D",
  "token_type": "Bearer",
  "expires_in": 3600,
  "refresh_token": "1/yJwnAH9IY5tqG98ENGtZD5E_R3SY005a4LbXpwaHwSQ"
}
```

sesiunea curentă e păstrată
într-un jeton (*token*) JSON

de experimentat accesul la diverse date (e.g., persoane de contact
via Google Contacts API) pe baza OAuth 2.0 Playground
developers.google.com/oauthplayground/

(în loc de) pauză

DÉJÀ VU



autentificare

Metode de autentificare:

bazate pe sesiunea Web via SID (*Session Identifier*)

implicit, Web-ul e *stateless*

autentificare

Metode de autentificare:
folosind jetoane (*tokens*)

token based authentication

autentificare

Metode de autentificare:
fără parolă (*passwordless*)

one-time-use URL

exemplu: Tumblr

Send me a magic link

Use password to log in

autentificare

Metode de autentificare:

SSO (*Single Sign-On*)

autentificarea utilizatorilor
în cadrul mai multor aplicații înrudite

autentificare

Metode de autentificare:

autentificare socială via alte conturi de utilizator

de exemplu, via rețele sociale
sau alte situri Web de încredere

autentificare

Metode de autentificare:

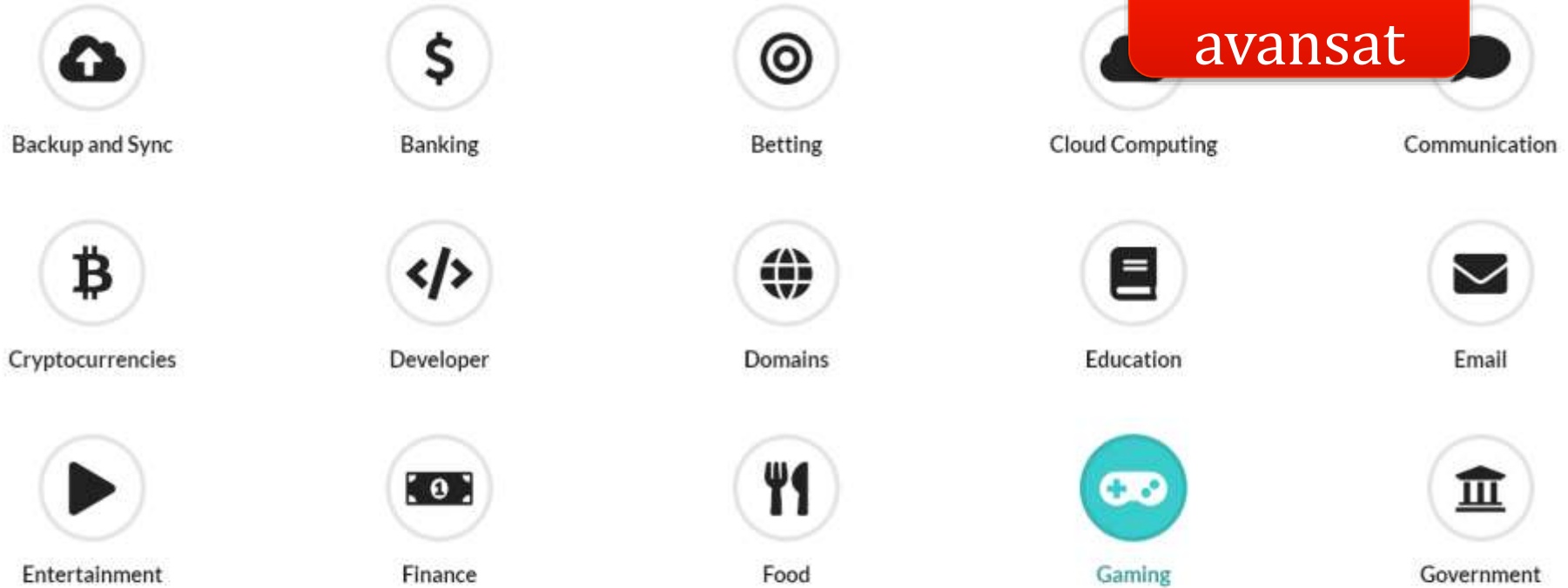
2FA (*Two-Factor Authentication*)

utilizatorul trebuie să ofere minim 2 probe (*factors*)
referitoare la identitatea sa

cunoștințe ori obiecte deținute exclusiv de acea persoană

de studiat și TOTP (*Time-Based One-Time Password Algorithm*) – RFC 6238: tools.ietf.org/html/rfc6238

avansat



Gaming	Docs	SMS	Phone Call	Email	Hardware Token	Software Token
 Ankama				✓		✓
 Aternos						✓
 Black Desert Online	Tell them to support 2FA on Twitter on Facebook					

autenticare multi-factor (2FA): twofactorauth.org

 Blizzard	⚠		✓		✓	✓
--	---	---	---	--	---	---

autentificare

Metode de autentificare:

autentificare biometrică

bazate pe amprentă (*fingerprint recognition*)

recunoașterea facială sau a unor organe

scanarea ochiului (iris, retină)

identificare vocală

analizarea codului genetic (*DNA matching*)

www.biometricsinstitute.org

autentificare

Metode de autentificare:

autentificare via dispozitiv hardware

exemplu tipic: *smartcard*

autentificare

Metode de autentificare – privire de ansamblu:

bazate pe sesiunea Web
folosind jetoane (*token based authentication*)

fără parolă (*passwordless*)

SSO (*Single Sign-On*)

autentificare socială

2FA (*Two-Factor Authentication*)

autentificare biometrică

autentificare via dispozitiv hardware

hackernoon.com/how-do-you-authenticate-mate-f2b70904cc3a

openid

OpenID

manieră descentralizată de autentificare a utilizatorului
la nivel de Web pe baza paradigmei SSO

utilizatorul poate demonstra că deține un URL specific
menit a-l identifica *on-line* via un ofertant (serviciu)
de identitate digitală (*identity provider*)
e.g., folosind o aplicație Web socială

openid

Fiecare identitate a unui utilizator e desemnată de un URL
(stabilit de *identity provider*)

exemplu: steamcommunity.com/openid/id/steamid

pentru a-și confirma identitatea,
utilizatorul va trebui să se autentifice:
nume de cont + parolă, *smart card*, date biometrice,...

openid

OpenID Connect

oferă un nivel vizând identitatea utilizatorului
(*identity layer*) pe baza protocolului OAuth 2

formatul de date folosit: **JWT** – *JSON Web Token*
standardizat de RFC 7519

openid.net/connect/

openid

OpenID Connect

biblioteci *open source* disponibile
pentru C, C#, Java, JavaScript, PHP, Python, Ruby,...
openid.net/developers/libraries/

openid

OpenID Connect

suport oferit de serverul Web:

mod_auth_openidc – modul Apache

github.com/pingidentity/mod_auth_openidc

L. Crilly, *Authenticating API Clients with JWT* (2016)

www.nginx.com/blog/authenticating-api-clients-jwt-nginx-plus/

avansat

Mesaje – de autentificare sau interschimb de informații – vehiculate în format JWT

jeton JWT

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VyaWQiOiIyOSIsImRhdGUiOiIyMDE3LTExIiwiaWF0eSI6IjE5OTYwNjc3LTI0MTkxLTI0MTkxIiwidmVudCI6ImFtZSIsIm51dCkiOiJlbnQoImZhbmHnIfQ.dGVpMn9BwnQQkwEUnbH6XK-Ura--txKRpW6X4ys35yw



resurse de interes + instrumente
oferite de jwt.io

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

PAYLOAD: DATA

```
{
  "userid": "69",
  "date": "2017-10-11",
  "name": "Tuxy Pinguinnesscool",
  "admin": true,
  "student": false
}
```

VERIFY SIGNATURE

```
HMACSHA256(
    base64UrlEncode(header) + "." +
    base64UrlEncode(payload),
    secret
)
```

openid

În conjuncție cu JWT, a se considera specificațiile
JOSE – JSON Object Signing and Encryption:

specificarea semnăturilor digitale

JWS – JSON Web Signature: tools.ietf.org/html/rfc7515

definirea metodelor criptografice

JWE – JSON Web Encryption: tools.ietf.org/html/rfc7516

reprezentarea cheilor criptografice

JWK – JSON Web Key: tools.ietf.org/html/rfc7517

identificarea & înregistrarea algoritmilor criptografici

JWA – JSON Web Algorithms: tools.ietf.org/html/rfc7518

Există alternative privitoare la REST?

model de acces la date: **graphql**

Graph Query Language

“a query language for APIs and a runtime for fulfilling those queries with your existing data”

graphql.org

sub jurisdicția *Linux Foundation*
gql.foundation

model de acces la date: **graphql**

Graph Query Language

declarativ

inspirat de JSON (*JavaScript Object Notation*)

strict (*strong-typed*)

model de acces la date: **graphql**

Graph Query Language

sunt permise interogări (*queries*) – operații de citire –
și actualizări (*mutations*) – operații de alterare a datelor

model de acces la date: **graphql**

Graph Query Language

sunt permise **interogări** (*queries*) – operații de citire –
și **actualizări** (*mutations*) – operații de alterare a datelor

se oferă suport pentru a anticipa ce date
vor fi întoarse + structura acestora

model de acces la date: **graphql**

Graph Query Language

- răspunsul oferit include doar datele ce au fost solicitate
- ▶ îmbunătățirea performanței la nivel de client

model de acces la date: **graphql**

Graph Query Language

răspunsul oferit include doar datele ce au fost solicitate

- ▶ îmbunătățirea performanței la nivel de client

rezolvarea problemelor vizând *over/under fetching*
(preluare a mai multor sau prea puține date)

philsturgeon.uk/api/2017/01/24/graphql-vs-rest-overview/
nordicapis.com/is-graphql-the-end-of-rest-style-apis/

	GraphQL	REST
entitate	resursă	resursă
format	JSON	orice <i>Media Type</i> (MIME) frecvent: JSON
protocol	HTTP – uzual, adoptă convenții proprii	independent de protocol (uzual, HTTP)
cine decide ce date vor fi întoarse	clientul	serverul
puncte terminale (<i>endpoints</i>)	un singur punct terminal pentru a oferi date conexe, dacă au fost specificate relații între ele	puncte terminale multiple (independente)
tipuri de date	<i>strong</i> (tipuri declarate explicit)	<i>weak</i> (verificarea tipurilor de date nu e obligatorie)
relație client-server	<i>fat client—fat server</i>	<i>thin client—fat server</i>
documentare	autodescriptiv (<i>self-describing</i>)	necesită terțe soluții (<i>e.g.</i> , OpenAPI Specification)
viziune	limbaj de interogare, specificație, colecție de instrumente	stil arhitectural

GraphQL ca alternativă la dezvoltarea de servicii via paradigma

model de acces la date: **graphql**

Aspecte de considerat de către implementatori:

probleme de securitate

– *e.g.*, autentificare, autorizare, refuz al serviciilor

suportul pentru *caching* trebuie oferit explicit

managementul versiunilor

blog.pusher.com/rest-versus-graphql/

model de acces la date: **graphql**

Instrumente pentru dezvoltatori:

implementare de referință pentru server (Node.js)

GraphQL.js

graphql.org/graphql-js/

biblioteci disponibile pentru C, Go, Java, .NET,
PHP, Python, Ruby, Swift, Typescript,...

graphql.org/code/

model de acces la date: graphql

Instrumente pentru dezvoltatori:

de experimentat și Apollo – dev.apolldata.com

suport la nivel de client

la nivel de Web via JavaScript – e.g., Angular, Vue, Meteor

pentru aplicații bazate pe React,
a se folosi Relay (Modern) – facebook.github.io/relay/

soluții pentru Android (Java) și iOS (Swift)

model de acces la date: graphql

Instrumente pentru dezvoltatori:

de experimentat și Apollo – dev.apolldata.com

inclusiv o abordare arhitecturală bazată pe micro-servicii

GrAMPS (*GraphQL Apollo Microservice Pattern Server*)

folosită de IBM

github.com/gramps-graphql/gramps

model de acces la date: **graphql**

Instrumente pentru dezvoltatori:
de experimentat și *framework*-ul **Graphene**
implementări pentru limbajele de programare
JavaScript – [graphene-js.org](https://github.com/graphql-js)
+
Python – [graphene-python.org](https://github.com/graphql-python)

GraphiQL

▶

avansat

```
1 # furnizeaza date despre o planeta
2 query GetPlanet($id: ID) {
3   planet(planetID: $id) {
4     name
5     terrains
6     residentConnection {
7       residents {
8         name
9         birthYear
10        a
11      } name
12      mass
13    }
14  }
15 }
```

interogare

rezultate

variabilă (parametru)

```
{
  "id": 1
}
```

◀ Search Results

Planet

A large mass, planet or planetoid in the Star Wars Universe, at the time of 0 ABY.

IMPLEMENTS

Node

FIELDS

name: String

diameter: Int

rotationPeriod: Int

orbitalPeriod: Int

gravity: String

population: Int

climates: [String]

terrains: [String]

surfaceWater: Float

residentConnection(after: String, first: Int, before:

GraphiQL – interogare GraphQL interactivă
în navigatorul Web a unui API
aici, „Războiul Stelelor” – graphql.org/swapi-graphql/

primele 33 de mesaje (+meta-date vizând utilizatorii care le-au expus)

```
{
  twitter {
    search(q: "Web application development", count: 33, result_type: mixed)
  {
    user {
      screen_name
      name
      followers_count
    }
    text
    created_at
  }
}
```

```
{
  "user": {
    "screen_name": "JavascriptBot_",
    "name": "Javascript Flux",
    "followers_count": 20932
  },
  "text": "Web Application Development: Basic Concepts https://t.co/Ealt9T2d1c #javascript",
  "created_at": "Sun Oct 22 04:05:17 +0000 2017"
},
{
  "user": {
    "screen_name": "DavidScott1087",
```

exemple de interogări via GraphQL ale unor API-uri publice
(*e.g.*, Giphy, Hacker News, Reddit): www.graphqlhub.com
în acest caz, Twitter



conceptul **Repository**
(depozit de cod-sursă)
și proprietățile aferente

```

1 # Getting data about a user + his/her first repo
2 query {
3   viewer {
4     name
5     createdAt
6     isHireable
7     repositories (first: 1) {
8       nodes {
9         description
10        isFork
11        isPrivate
12        lan

```

languages

lockReason

primaryLanguage

defaultBranchRef

QUERY

LanguageConnection A list containing a breakdown of the language composition of the repository.

1 {}

```

{
  "data": {
    "viewer": {
      "name": "Sabin Buraga"
      "createdAt": "2015-04-
      "isHireable": false,
      "repositories": {
        "nodes": [
          {
            "description":
(databases-ontology)",
            "isFork": false
            "isPrivate": fa
          }
        ]
      }
    }
  }
}

```

< Query

Repository

isMirror: Boolean!

isPrivate: Boolean!

issue(number: Int!): Issue

issueOrPullRequest(number: Int!):
IssueOrPullRequest

issues(first: Int, after: String, last: Int, before:
String, labels: [String!], orderBy: IssueOrder,
states: [IssueState!]): IssueConnection!

label(name: String!): Label

labels(first: Int, after: String, last: Int, before:
String): LabelConnection

languages(first: Int, after: String, last: Int,
before: String, orderBy: LanguageOrder):
LanguageConnection

license: String (DEPRECATED)

licenseInfo: License

testarea interactivă a API-ului GitHub
implementat via GraphQL
developer.github.com/v4/

model de acces la date: **graphql**

Alte exemplificări:

AWS AppSync – sincronizare în timp-real a datelor
via servicii Amazon disponibile „în nori”

docs.aws.amazon.com/appsync/latest/devguide/

model de acces la date: **graphql**

Alte exemplificări:

Shopify – acces la date vizând comerțul electronic

help.shopify.com/en/api/custom-storefronts/storefront-api/

model de acces la date: **graphql**

Alte exemplificări:

Yelp – acces la recenzii de produse/servicii

www.yelp.com/developers/graphql/guides/intro

model de acces la date: graphql

Resurse de interes, studii de caz, noutăți:

Awesome GraphQL

github.com/chentsulin/awesome-graphql

Open GraphQL

medium.com/open-graphql

model de acces la date: **graphql**

Alternativă:

YQL (Yahoo! Query Language)

abstractizează accesul la surse de date eterogene
ce pot fi obținute via servicii Web

developer.yahoo.com/yql/

model de acces la date: graphql

Alternativă:

TreeQL

inspirat de REST și GraphQL

răspunsul la o interogare reprezintă un „arbore” de obiecte JSON corespunzătoare structurii (relațiilor) bazei de date relaționale (SQL) interogate via REST

treeql.org

model de acces la date: **graphql**

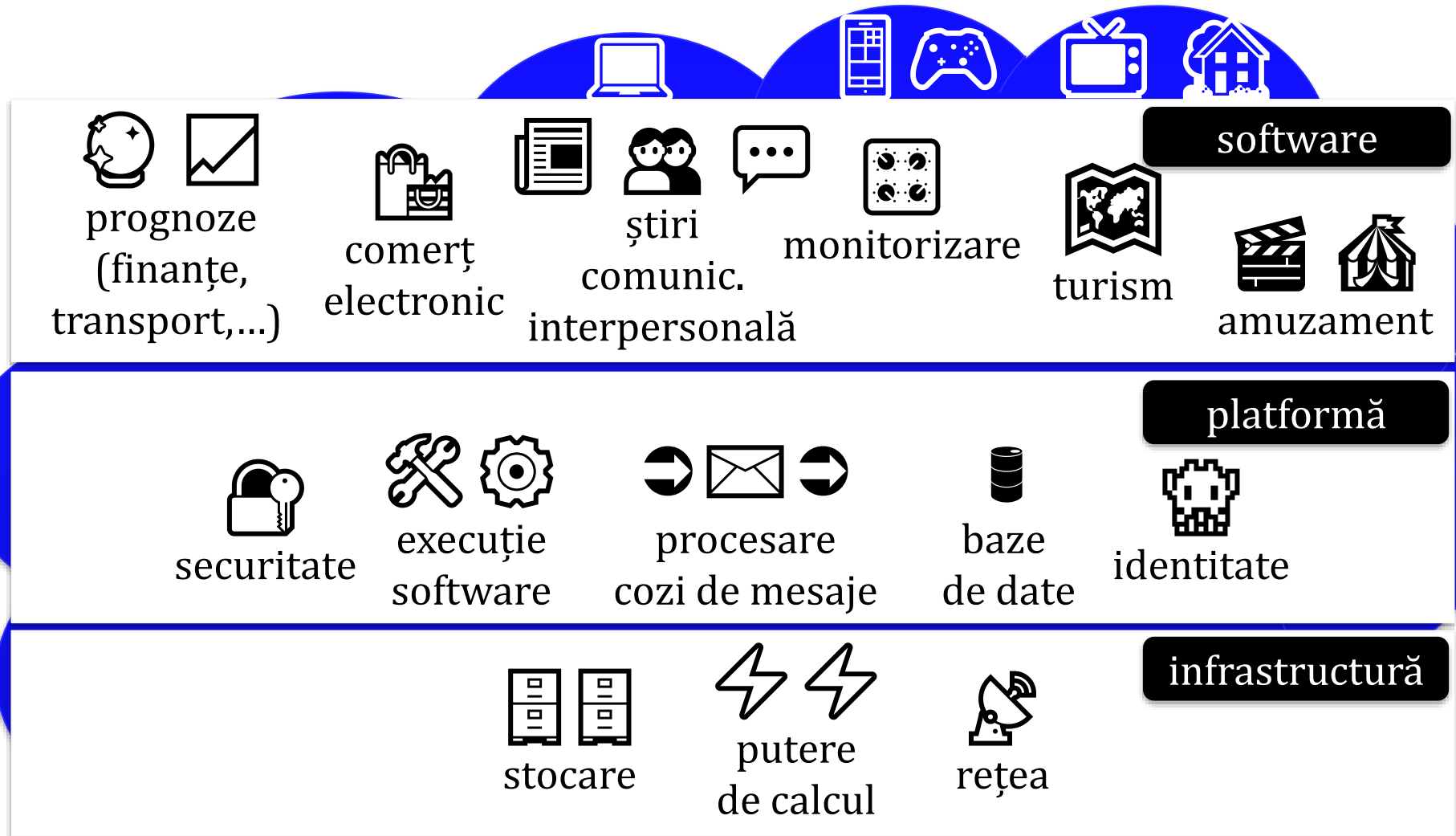
Alternativă:

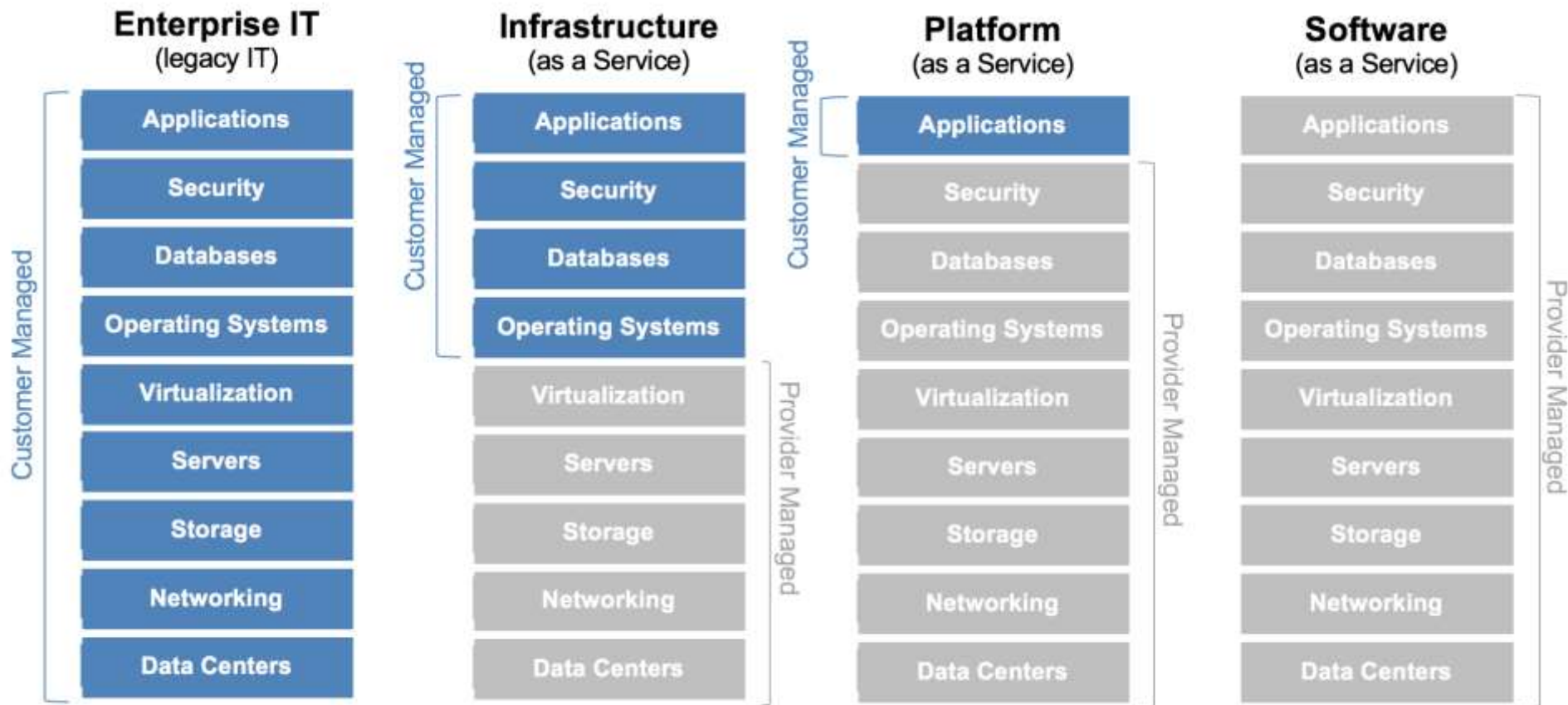
TreeQL

implementare de referință în PHP7 – PHP-CRUD-API:
github.com/mevdschee/php-crud-api

alte soluții (C#, Go, Java, Node.js, Python):
treeql.org/code/

Web – ingredient cheie al tehnologiilor în „nori” *cloud computing*



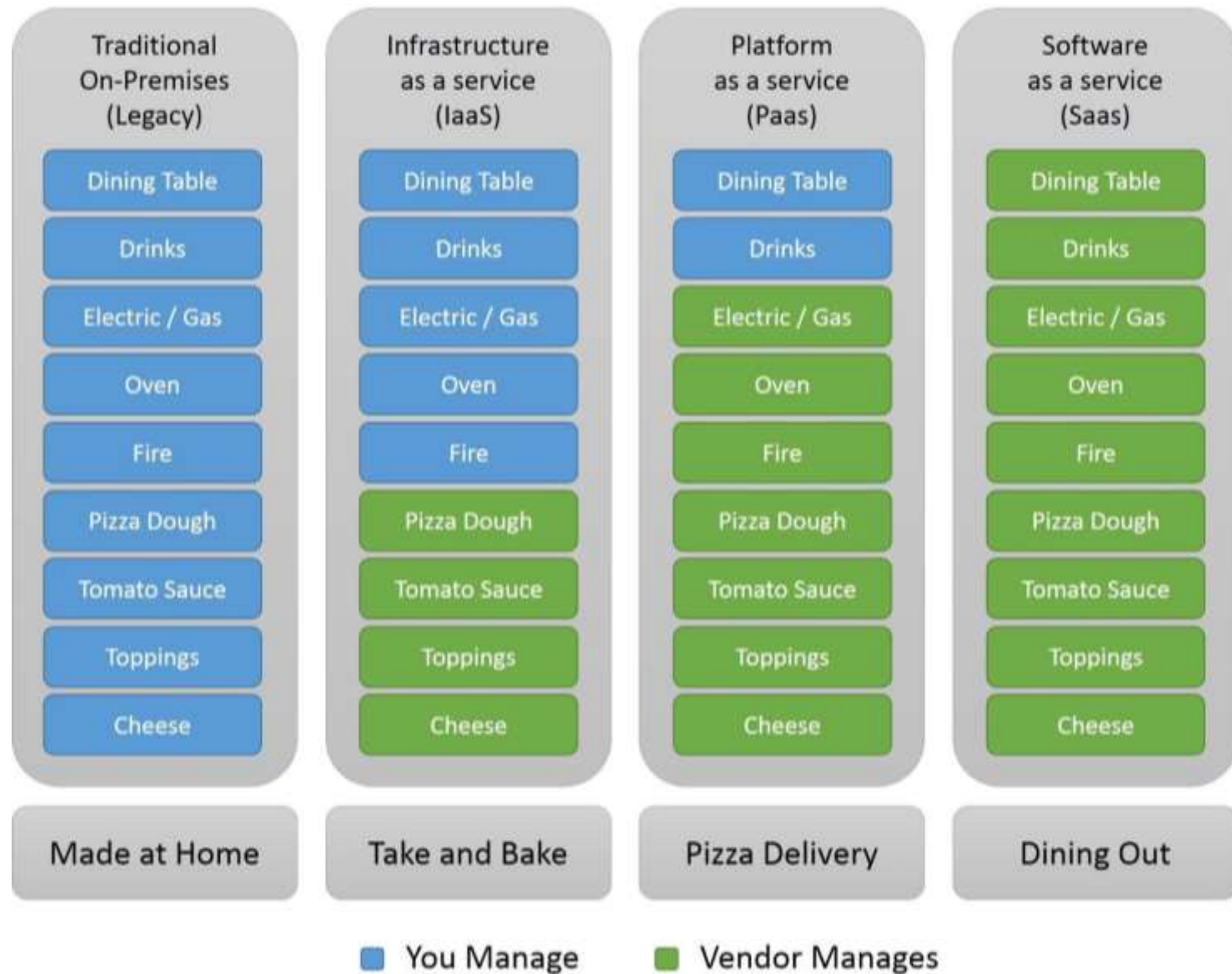


anumite funcționalități pot fi gestionate
„în propria ogradă” (*on-premises*)
sau de un furnizor de servicii disponibile „în nori”

conform (Eizadirad, 2017)

www.linkedin.com/pulse/iaas-paas-saas-explained-compared-arsalan-eizadirad

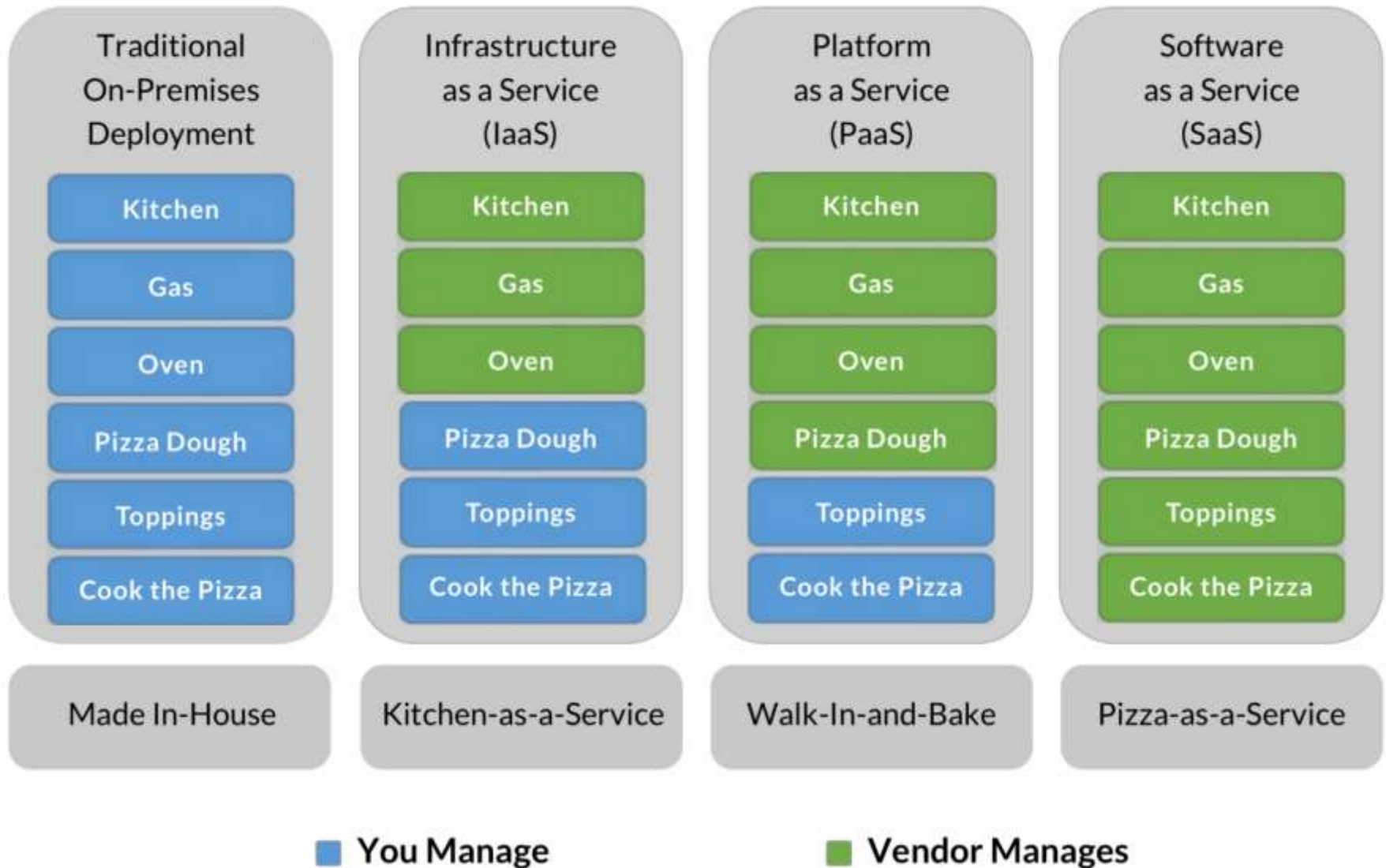
Pizza as a Service



A. Barron, *Pizza As A Service* (2014)

www.linkedin.com/pulse/20140730172610-9679881-pizza-as-a-service

New Pizza as a Service



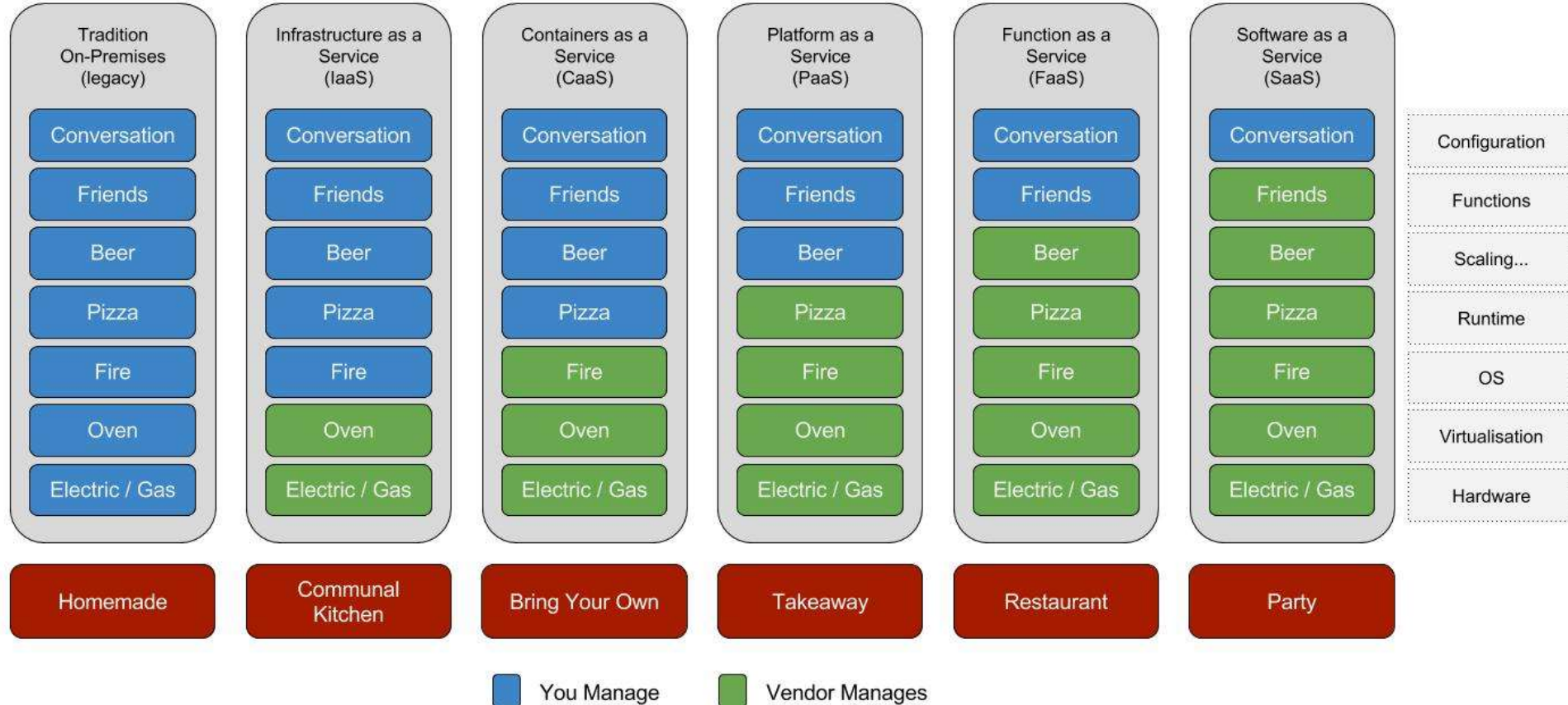
D. Ng, *SaaS, PaaS and IaaS explained in one graphic* (2017)

m.oursky.com/saas-paas-and-iaas-explained-in-one-graphic-d56c3e6f4606



Pizza as a Service 2.0

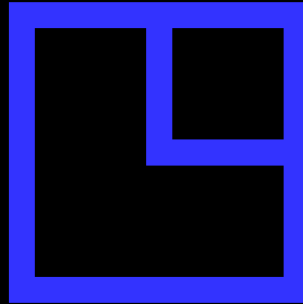
<http://www.paulkerrison.co.uk>



P. Kerrison, *Pizza As A Service 2.0* (2017)
www.paulkerrison.co.uk/random/pizza-as-a-service-2-0

rezumat

dezvoltare de servicii Web



de la micro-servicii și specificarea API-urilor
la *serverless* și accesul la date via GraphQL

🗑️ All HTML CSS JS **XHR** Fonts Images Media Flash WS Other ❑ Persist Logs ❑ Disable cache

Status	Method	File	Domain	Cause	Type	Transferr
● 200	POST	<input type="checkbox"/> /_save/	🔒 jsfiddle.net	📄 xhr	json	311 B

🕒 One request | 87 B / 311 B transferred | Finish: 1.71 min | DOMContentLoaded: 2.53 s | load: 2.31 s

🗑️ 🔍 Filter output

- ▶ GET https://docs.google.com/static/forms/client/css/2785006849-formview_embedded_st_ltr.css
- ▶ GET https://docs.google.com/static/forms/client/js/1918813819-formviewer_prd.js
- ⚠️ Use of getPreventDefault() is deprecated. Use defaultPrevented instead.
- ▶ GET https://togetherjs.com/togetherjs/images/button-share-active.png
- ▶ GET https://togetherjs.com/togetherjs/images/icon-close-active.png
- 📩 Send: ▶ Object { type: "form-focus", element: "#savenew" }
- 📩 Send: ▶ Object { type: "cursor-click", element: "#savenew", offsetX: 40.366668701171875, offsetY: 31 }
- ▼ POST XHR https://jsfiddle.net/_save/

Headers	Cookies	Params	Response	Timings
🔍 Filter properties				
▼ JSON				
<pre> pastie_url_relative: /busaco/akq5f1ht/ slug: akq5f1ht shell_id: 317850410 </pre>				
▼ Response payload				
1 ▼ {"pastie_url_relative": "/busaco/akq5f1ht/", "slug": "akq5f1ht", "shell_id": 317850410}				

▶ GET https://jsfiddle.net/busaco/akq5f1ht/embedded/

episodul viitor:
transfer asincron al datelor
aplicații Web hibride (*mash-ups*)