

TECHNOLOGIE ORAZ PROTOKOŁY UMOŻLIWIAJĄCE INTEGRACJĘ APLIKACJI MOBILNYCH Z SERWISAMI INTERNETOWYMI

BARTŁOMIEJ POKUTYCKI

PYTANIE SPECJALNOŚCIOWE NR 13

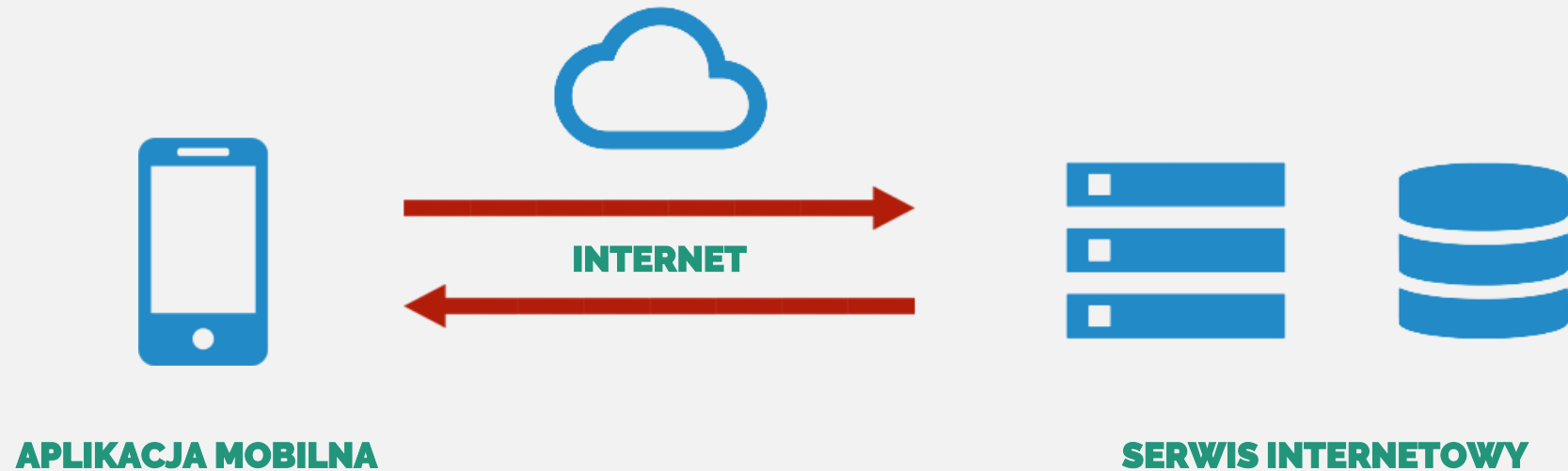


Politechnika Wrocławska

Serwis internetowy

„System zaprojektowany w celu obsługi interakcji pomiędzy maszynami w sieci”

W3C



Protokół HTTP

Hypertext Transfer Protocol

Umożliwia komunikację pomiędzy serwisami, a klientami

Protokół bezstanowy

Wiadomości: żądanie i odpowiedź

1996: 1.0

1997: 1.1

2015: 2.0



Protokół HTTP

method	path	protocol
GET	/tutorials/other/top-20-mysql-best-practices/	HTTP/1.1

```
Host: net.tutsplus.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Cookie: PHPSESSID=r2t5uvjq435r4q7ib3vtdjq120
Pragma: no-cache
Cache-Control: no-cache
```

GET, POST, PUT, DELETE

1xx – kody informacyjne

2xx – kody powodzenia

3xx – kody przekierowania

4xx – kody błędu (aplikacji)

5xx – kody błędu (serwera)

SOAP



Simple Object Access Protocol



Zaprojektowany w 1998 roku, standard W3C od 2003 roku



Możliwość wykorzystania protokołu HTTP do transmisji danych



Określona struktura wysyłanych wiadomości z wykorzystaniem XML

SOAP - XML

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:ns="http://dpd.com/common/service/types/LoginService/2.0">
  <soapenv:Header/>
  <soapenv:Body>
    <ns:getAuth>
      <delisId>*userid*</delisId>
      <password>*password*</password>
      <messageLanguage>de_DE</messageLanguage>
    </ns:getAuth>
  </soapenv:Body>
</soapenv:Envelope>
```

return:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns2:getAuthResponse xmlns:ns2="http://dpd.com/common/service/types/LoginService/2.0"
      xmlns:ns3="http://dpd.com/common/service/exceptions">
      <return>
        <delisId>*userid*</delisId>
        <customerUid>*userid*</customerUid>
        <authToken>LTM1NDQyNzk0MDY0OTc5ODE0NwRRMTMzMzEwODc2MTc0MARR</authToken>
        <depot>0163</depot>
      </return>
    </ns2:getAuthResponse>
  </soap:Body>
</soap:Envelope>
```

Uniwersalny język znaczników

Format tekstowy

Specyfikacja W3C

SOAP - WSDL

```
<wsdl:definitions targetNamespace="http://math.example.com" name="MathFunctionsDef">
  <wsdl:message name="addIntResponse">
    <wsdl:part name="addIntReturn" type="xsd:int" />
  </wsdl:message>
  <wsdl:message name="addIntRequest">
    <wsdl:part name="a" type="xsd:int" />
    <wsdl:part name="b" type="xsd:int" />
  </wsdl:message>
  <wsdl:portType name="AddFunction">
    <wsdl:operation name="addInt" parameterOrder="a b">
      <wsdl:input message="impl:addIntRequest" name="addIntRequest" />
      <wsdl:output message="impl:addIntResponse" name="addIntResponse" />
    </wsdl:operation>
  </wsdl:portType>
  <service name="MathFunctions"/>
</wsdl:definitions>
```

Język definiowania usług sieciowych

Wykorzystuje XML

Opis może zostać umieszczony w UDDI

REST API



Representational State Transfer



Styl architektoniczny zaproponowany w 2000 roku



Wykorzystanie HTTP do transmisji danych



Jednokierunkowa komunikacja klient-serwer

REST API

GET	/projects
GET	/projects/:id
GET	/projects/:id/tasks
POST	/projects
GET	/tasks
GET	/tasks/:id
POST	/tasks

AKCJA

CREATE
READ
UPDATE
DELETE

METODA HTTP

POST
GET
PUT
DELETE

REST API - JSON

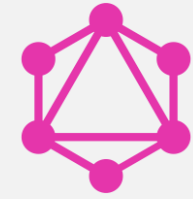
```
{  
  "id": 1252,  
  "name": "John",  
  "address": { ... },  
  "birthday": "July 21, 1985",  
  "tasks": [  
    {  
      "id": 158,  
      "title": "Lorem",  
      "content": "Lorem ipsum ..."  
    },  
    {  
      "id": 162,  
      "title": "Lorem 2",  
      "content": "Lorem ipsum ..."  
    }  
  ],  
  ...  
}
```

Format tekstowy

Podzbiór języka JavaScript

Niezależny od konkretnego języka

GraphQL i Falcor



Umożliwiają wybór wyłącznie niezbędnych danych

**Jedna specyfikacja,
mnogość implementacji**

Jeden endpoint

Język zapytań

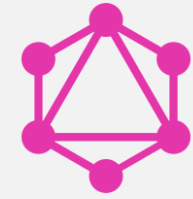
Query & Mutation

Biblioteka JavaScript

Virtual JSON Graph

Operowanie na lokalnych obiektach

GraphQL i Falcor



```
GraphiQL ▶ Prettify < Docs

1 {
2   allStarships(
3     first: 2
4     after: "YXJyYXljb25uZWN0aW9uOjI="
5   ) {
6     pageInfo {
7       hasNextPage
8     }
9     edges {
10      cursor
11      node {
12        id
13        name
14      }
15    }
16  }
17 }
```

```
{
  "data": {
    "allStarships": {
      "pageInfo": {
        "hasNextPage": true
      },
      "edges": [
        {
          "cursor": "YXJyYXljb25uZWN0aW9uOjM=",
          "node": {
            "id": "c3RhcnNoaXBzOjk=",
            "name": "Death Star"
          }
        },
        {
          "cursor": "YXJyYXljb25uZWN0aW9uOjQ=",
          "node": {
            "id": "c3RhcnNoaXBzOjEw",
            "name": "Millennium Falcon"
          }
        }
      ]
    }
  }
}
```

QUERY VARIABLES

```
class Model {
  get(...PathSet): ModelResponse
}

var falcor = require('falcor' 2.0.5 );
var HttpDataSource = require('falcor-http-datasource' 0.1.3 );
var baseUrl = 'https://falcor-server-sample-nvutywuvbl.now.sh';

var model = new falcor.Model({
  source: new HttpDataSource(baseUrl + '/model.json')
});

var jsonGraph = await model.get(["todos", {from: 0, to:1}, "name"], ["todos",
"length"]);
```

Save on RunKit Node 10 help ▶ run

```
Object
└─ json: Object
  └─ todos: Object
    ├── 0: Object { $__path: 0, name: "get milk from corner store" }
    ├── 1: Object { $__path: 1, name: "withdraw money from ATM" }
    └─ $__path: ["todos"]
    # length: 2
```

Odzwierciedlenie struktury zapytania

Zwrócenie wybranej części zdefiniowanego grafu

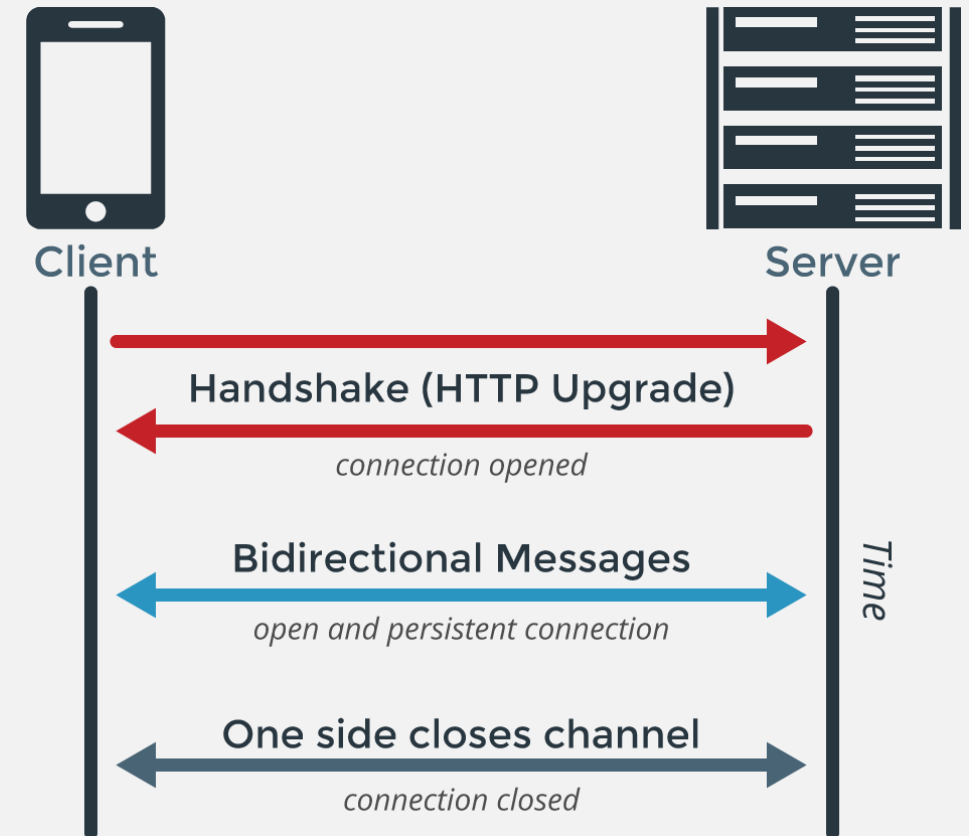
WebSocket

Wykorzystanie protokołów TCP i HTTP

URI – ws: i wss:

Dwukierunkowy kanał komunikacyjny

Dowolność formatu danych



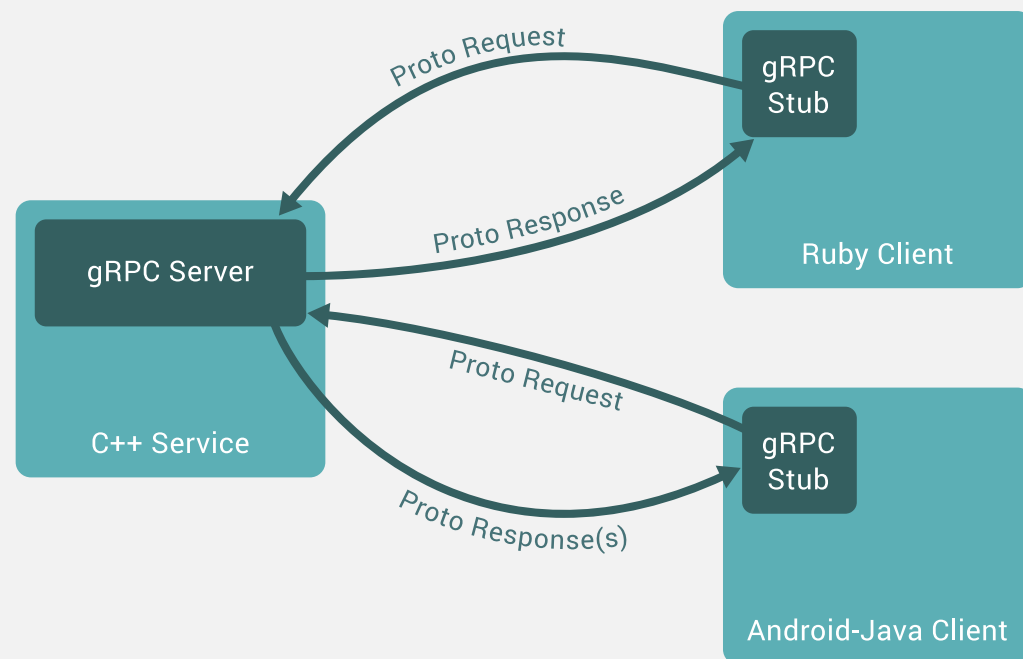
gRPC

Nowoczesny framework RPC

Wykorzystanie HTTP/2

Dwa rodzaje: Unary oraz Streaming

IDL: Protocol Buffers



gRPC – Protocol Buffers

```
syntax = "proto3";  
package time;  
  
service Time {  
  rpc GetTime (TimeRequest) returns (TimeReply) {}  
}  
  
// Empty Request Message  
message TimeRequest {  
}  
  
// The response message containing the time  
message TimeReply {  
  string message = 1;  
}
```

Annotations:

- Service name: `Time`
- RPC Input Type: `TimeRequest`
- RPC Output Type: `TimeReply`
- Define RPC call: `rpc GetTime (TimeRequest) returns (TimeReply) {}`
- It's a good idea to create a type even if it's empty: `message TimeRequest { }`
- Our RPC call will time in string format: `string message = 1;`

Serializacja danych

Stubs

Push Notification

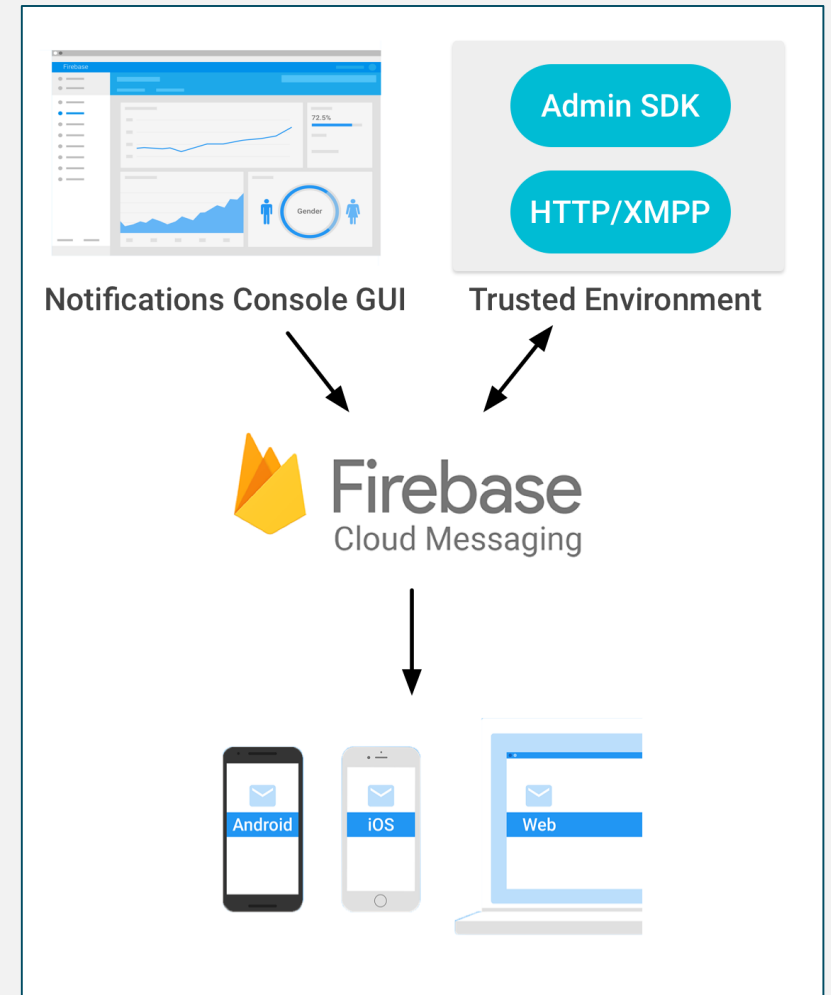
Małe wiadomości

Mogą dotrzeć do klienta w dowolnym czasie

Wykorzystanie serwera pośredniczącego

HTTP: cloud-to-device

XMPP: device-to-cloud, cloud-to-device



Push Notification



Notification message



Data message

Literatura




W3C, *Web Services Architecture*, <https://www.w3.org/TR/ws-arch> [dostęp: 04.05.2019]



REST API Tutorial, <https://restfulapi.net> [dostęp: 04.05.2019]



Facebook Open Source, *GraphQL: A query language for your API*, <https://graphql.org> [dostęp: 04.05.2019]



Netflix Open Source, *Falcor: One Model Everywhere*, <https://netflix.github.io/falcor> [dostęp: 04.05.2019]



Ilya Grigorik, *WebSocket*, <https://hpbn.co/websocket> [dostęp: 04.05.2019]



Google Developers, *Firebase Cloud Messaging*, <https://firebase.google.com/docs/cloud-messaging> [dostęp: 04.05.2019]



gRPC, <https://grpc.io> [dostęp: 04.05.2019]

Źródła obrazów

https://commons.wikimedia.org/wiki/File:GraphQL_Logo.svg

<https://netflix.github.io/falcor/images/falcor-logo-twitter-card.png>

https://www.flaticon.com/free-icon/smartphone_65680

https://cdn.tutsplus.com/net/uploads/legacy/511_http/request_header.png

https://www.researchgate.net/profile/Eran_Toch/publication/265265754/figure/fig2/AS:669542653124618@1536642816584/An-example-of-a-WSDL

https://docs.microsoft.com/pl-pl/media/common/i_http.svg

https://cdn-images-1.medium.com/max/2400/1*30sHDt1YQzUH5lc95hizRQ.png

<https://i1.wp.com/www.kisiel.ovh/wp-content/uploads/2015/11/WebSockets-Diagram.png>

<https://docs.microsoft.com/pl-pl/xamarin/android/data-cloud/google-messaging/firebase-cloud-messaging-images/02-app-registration-sml.png>

<https://firebase.google.com/docs/cloud-messaging/images/messaging-overview.png>

https://cdn-images-1.medium.com/max/800/0*ooiTFOtAOHiPlWAA.png

TECHNOLOGIE ORAZ PROTOKOŁY UMOŻLIWIAJĄCE INTEGRACJĘ APLIKACJI MOBILNYCH Z SERWISAMI INTERNETOWYMI

BARTŁOMIEJ POKUTYCKI

PYTANIE SPECJALNOŚCIOWE NR 13



Politechnika Wrocławska