REST NA PRAVI NAČIN

Hrvoje Crnjak
Pet minuta d.o.o.

HOW RESTFUL IS MY REST ?

Krenimo od početka...

- HTTP/0.9 (1991)
  - Protokol ne zadovoljava potrebe
- Kako odrediti što nedostaje?
  - Potrebno definirati model kako bi Web *trebao* raditi
- Odgovor
  - **“HTTP object model”**
  - Zbog imena pogrešno tumačen kao model implementacije HTTP servera

I bi REST

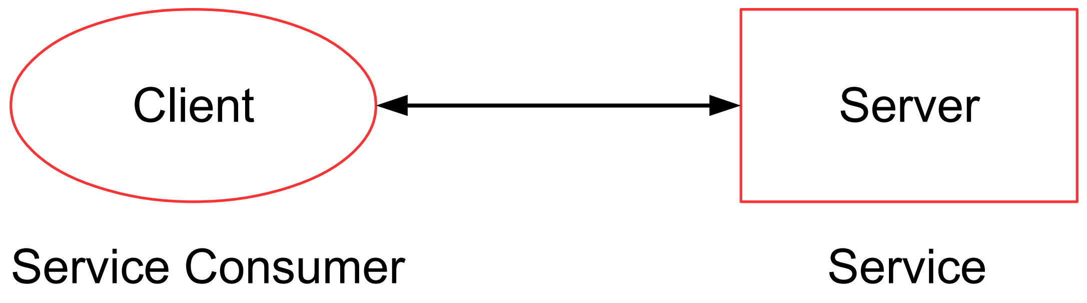
- REpresentational State Transfer
- Odgovor na potrebu za modelom Weba
- Razvijen od strane grupa za razvoj HTTP protokola
  - Glavni kreator Roy Thomas Fielding
- Prva primjena
  - HTTP/1.0 (1994)
- Objavljen
  - Doktorska disertacija Roya Fieldinga (2000)

REST službeno

- Koordinirani skup arhitekturalnih ograničenja s ciljem minimiziranja latencije i generalne mrežne komunikacije, te maksimiziranja nezavisnosti i skalabilnosti komponenata implementacije
- 6 ograničenja (constraint) čini REST stil arhitekture

Client-Server

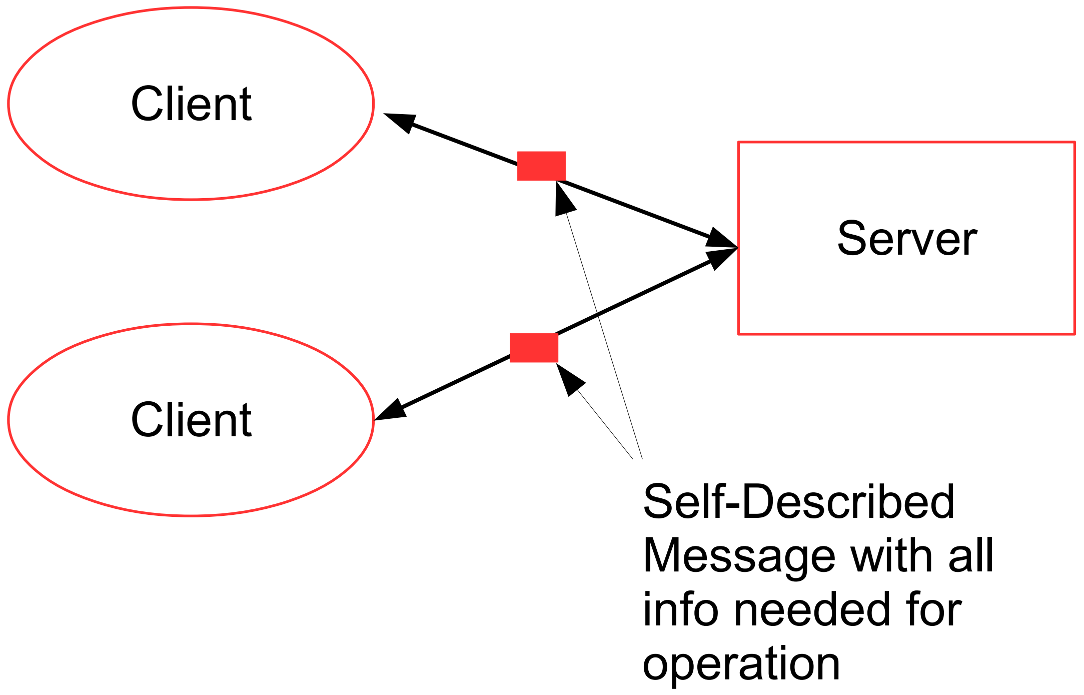
- Separation of concerns
- Rezultat
  - Nezavisan razvoj komponenata
  - Komponente pojednostavljene



Stateless

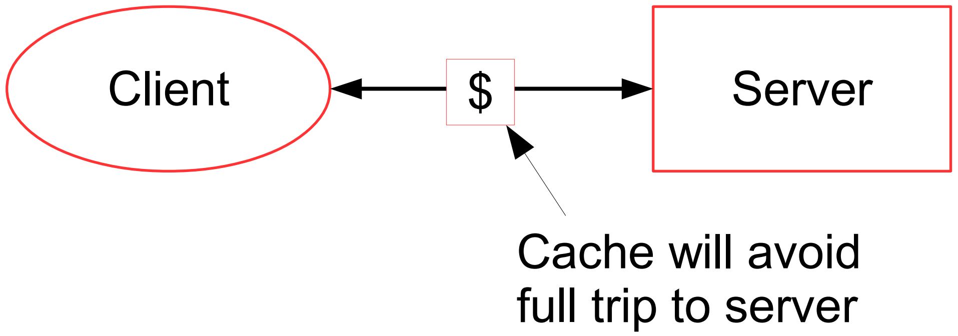
- Server ne treba pamtiti stanje aplikacije
- Rezultat
  - Klijent u svakom requestu šalje sve informacije potrebne za izvršavanje istoga
- Posljedice
  - Poboljšana efikasnost monitoring sustava
  - Jednostavniji recovery
  - Poboljšana skalabilnost servera (nema potrebe za pohranom stanja)
  - Session state u potpunosti čuvan na klijentu
  - Network overhead

Stateless


Client
Client
Server
Self-Described Message with all info needed for operation

Cache

- Podaci responsea označavaju su kao cacheable ili non-cacheable
- Posljedice
  - Poboljšana efikasnost, skalabilnost, te performanse iz korisničke perspektive
  - Smanjena pouzdanost (što ako je podatak zastario)

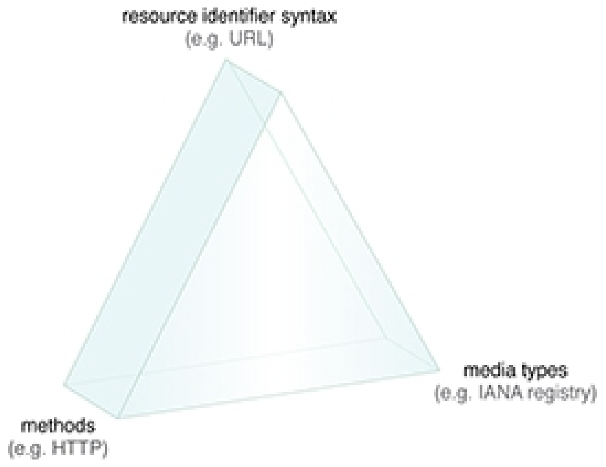


Uniform Interface

- Sve komponente arhitekture moraju dijeliti isto **tehničko** sučelje
  - Sučelje dovoljno generičko za primjenu na široki raspon domena
- Prednost
  - Sve komponente arhitekture “znaju” jezik komunikacije
- Mana
  - Zbog standardiziranosti sučelja, smanjena efikasnost

Uniform Interface

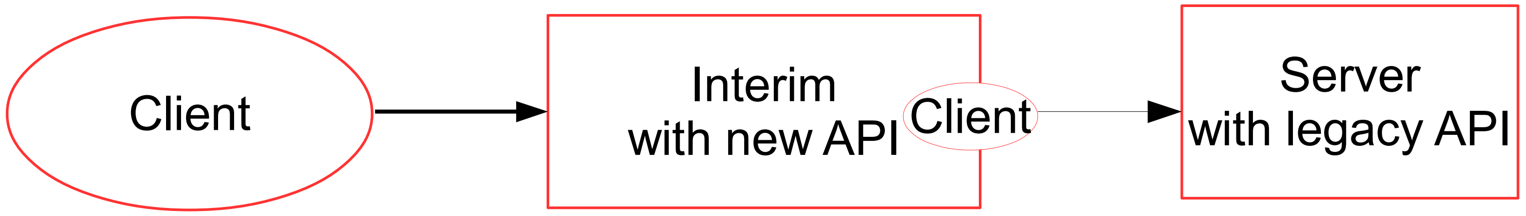
- Zahtjevi na izgled korisničkog sučelja :
  - Identification of resources
  - Manipulation of resources through representations
  - Self descriptive messages
  - Hypermedia as the engine of application state (HATEOAS)



Layered System

- Rješenje se može sastojati od više slojeva arhitekture
  - Sloj ne može “vidjeti” dalje od idućega u “nizu”
  - Manipulacija slojevima prema potrebi
- Slučajevi uporabe
  - Enkapsulacija legacy sustava
  - Load balancing, shared cache
  - Analiza i transformacija poruka
- U praksi
  - Proxy, gateway

Layered System



Code on Demand

- Dozvoljava klijentu da dohvati i izvrši kod sa servera
- Bazira se na web-based tehnologijama
  - Browser plug-ins
  - Applets
  - Client-side scripting languages (JavaScript)
- OPCIONALNO!

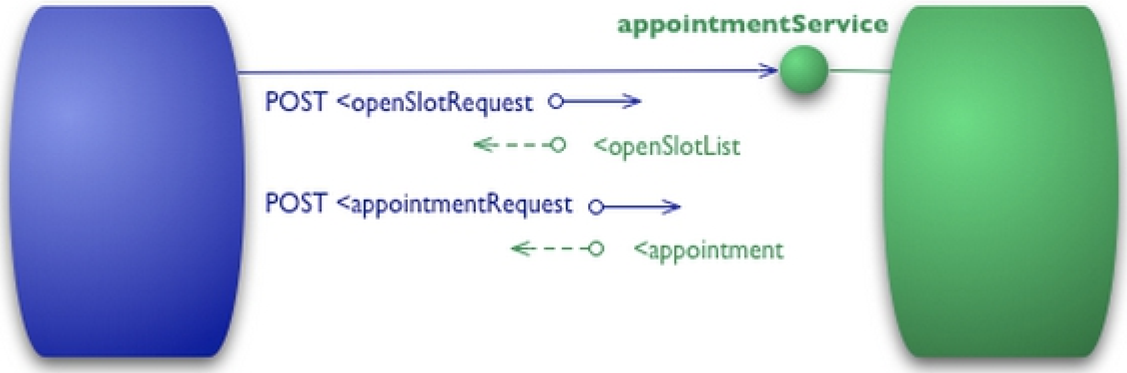
Richardson Maturity Model

- Model za ocjenjivanje korisničkog API-ja prema tome kako implementira zahtjeve REST arhitekture
- Autor
  - Leonard Richardson
- Objavljen
  - QCon konferencija 2008. god.
- 4 levela (0 – 3)

Level 0 : The Swamp of POX

- Jedan URI
  - Definira sve operacije (cjelokupnu funkcionalnost)
  - Željena operacija navodi se u samoj poruci
- Jedan HTTP glagol
  - Najčešće POST
- HTTP isključivo kao transportni protokol
  - Tunneling mehanizam za vlastiti remote interaction sustav (pr. SOAP)

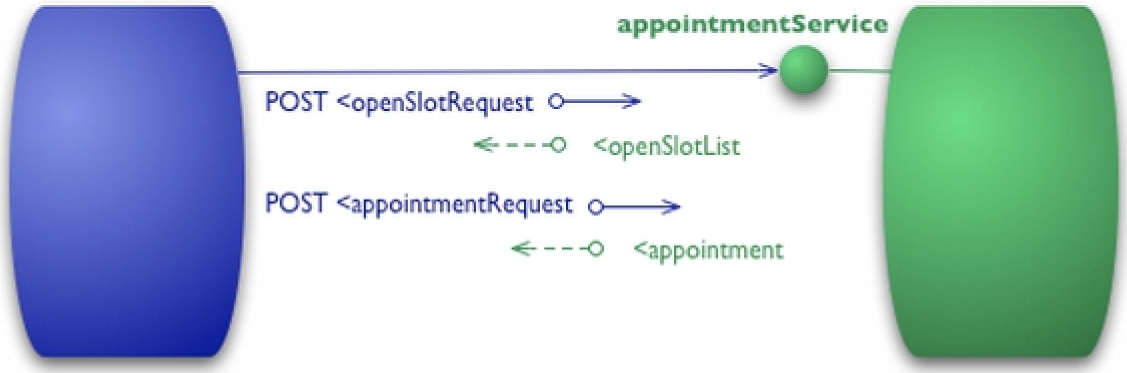
Level 0: The Swamp of POX



```
POST /appointmentService HTTP/1.1
[various other headers]

<openSlotRequest date = "2010-01-04" doctor = "mjones"/>
```

```
HTTP/1.1 200 OK
[various headers]

<openSlotList>
  <slot start = "1400" end = "1450">
    <doctor id = "mjones"/>
  </slot>
  <slot start = "1600" end = "1650">
    <doctor id = "mjones"/>
  </slot>
</openSlotList>
```

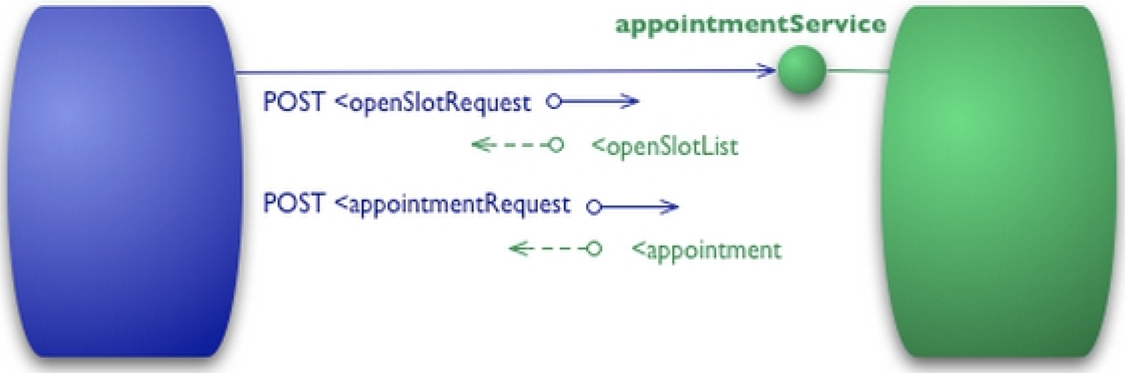
Level 0: The Swamp of POX



```
POST /appointmentService HTTP/1.1
[various other headers]

<appointmentRequest>
  <slot doctor = "mjones" start = "1400" end = "1450"/>
  <patient id = "jsmith"/>
</appointmentRequest>
```

```
HTTP/1.1 200 OK
[various headers]

<appointment>
  <slot doctor = "mjones" start = "1400" end = "1450"/>
  <patient id = "jsmith"/>
</appointment>
```

Level 0: The Swamp of POX



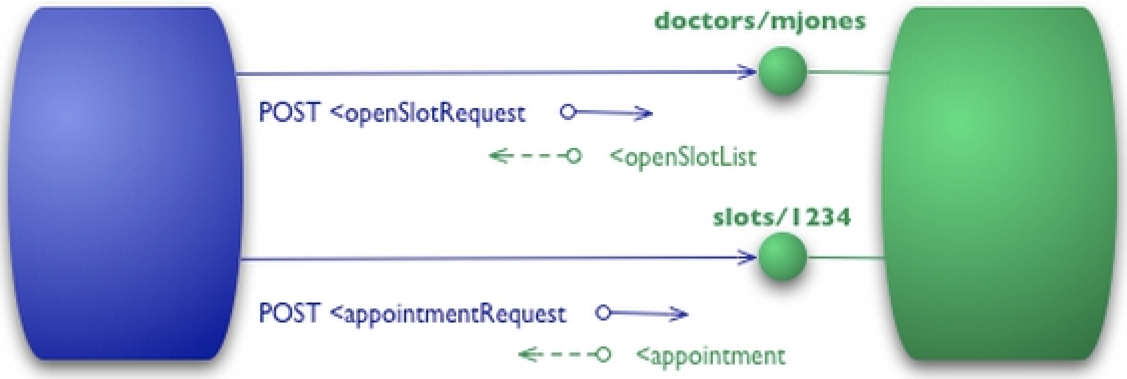
```
POST /appointmentService HTTP/1.1
[various other headers]

<appointmentRequest>
  <slot doctor = "mjones" start = "1400" end = "1450"/>
  <patient id = "jsmith"/>
</appointmentRequest>
```

```
HTTP/1.1 200 OK
[various headers]

<appointmentRequestFailure>
  <slot doctor = "mjones" start = "1400" end = "1450"/>
  <patient id = "jsmith"/>
  <reason>Slot not available</reason>
</appointmentRequestFailure>
```

Level 1 : Resources

- URI-ji definirani prema resursima aplikacije
- I dalje jedan HTTP glagol
- Divide and conquer metoda
  - Veliki service endpoint lomi u više manjih (jednostavnijih) resursa

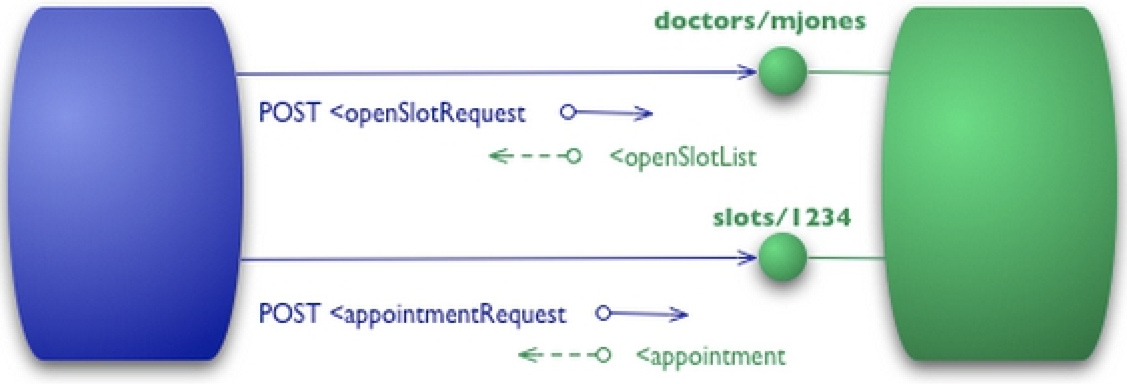
Level 1 : Resources



```
POST /doctors/mjones HTTP/1.1
[various other headers]

<openSlotRequest date = "2010-01-04"/>
```

```
HTTP/1.1 200 OK
[various headers]


<openSlotList>
  <slot id = "1234" doctor = "mjones" start = "1400" end = "1450"/>
  <slot id = "5678" doctor = "mjones" start = "1600" end = "1650"/>
</openSlotList>
```

Level 1 : Resources



```
POST /slots/1234 HTTP/1.1
[various other headers]

<appointmentRequest>
  <patient id = "jsmith"/>
</appointmentRequest>
```

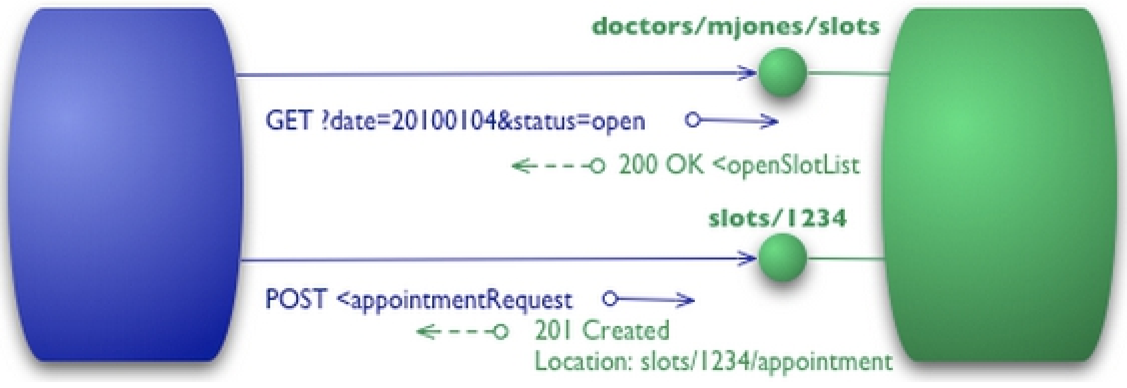
```
HTTP/1.1 200 OK
[various headers]

<appointment>
  <slot id = "1234" doctor = "mjones" start = "1400" end = "1450"/>
  <patient id = "jsmith"/>
</appointment>
```

Level 2 : HTTP Verbs

- Više URI-ja
- HTTP glagoli i statusi koriste se kako su definirani specifikacijom
- HTTP prestaje biti samo transportni protokol
  - Koristi se standardni set glagola i statusa kako bi se slične situacije rješavale na slične i predvidljive načine

    (why reinvent the wheel)

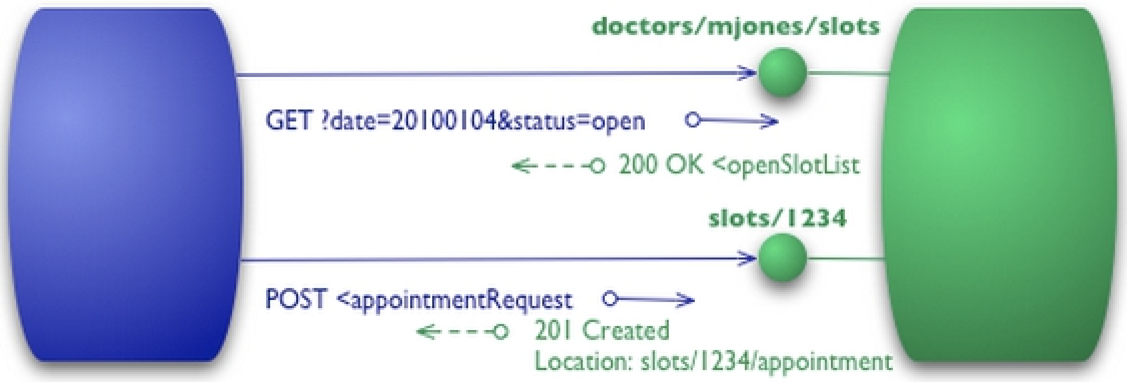
Level 2 : HTTP Verbs



```
GET /doctors/mjones/slots?date=20100104&status=open HTTP/1.1
Host: royalhope.nhs.uk
```

```
HTTP/1.1 200 OK
[various headers]

<openSlotList>
  <slot id = "1234" doctor = "mjones" start = "1400" end = "1450"/>
  <slot id = "5678" doctor = "mjones" start = "1600" end = "1650"/>
</openSlotList>
```

Level 2 : HTTP Verbs

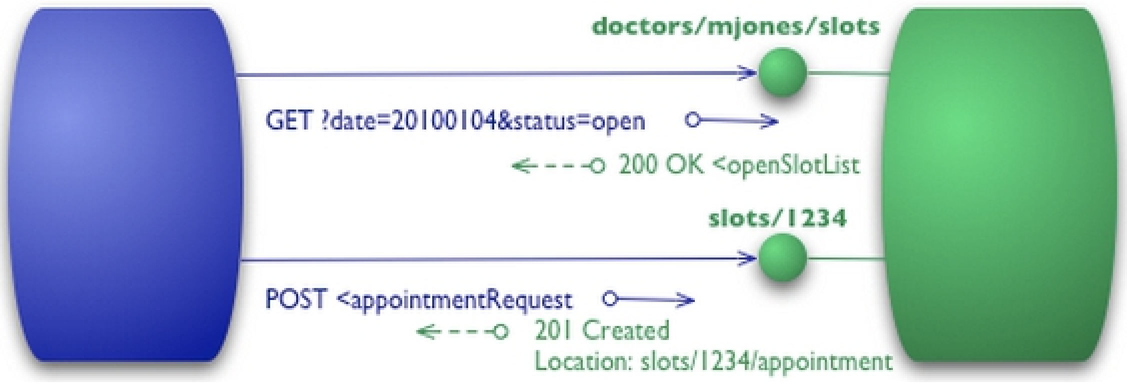


```
POST /slots/1234 HTTP/1.1
[various other headers]

<appointmentRequest>
  <patient id = "jsmith"/>
</appointmentRequest>
```

```
HTTP/1.1 201 Created
Location: slots/1234/appointment
[various headers]

<appointment>
  <slot id = "1234" doctor = "mjones" start = "1400" end = "1450"/>
  <patient id = "jsmith"/>
</appointment>
```

JavaCro

Level 2 : HTTP Verbs



```
POST /slots/1234 HTTP/1.1
[various other headers]

<appointmentRequest>
  <patient id = "jsmith"/>
</appointmentRequest>
```

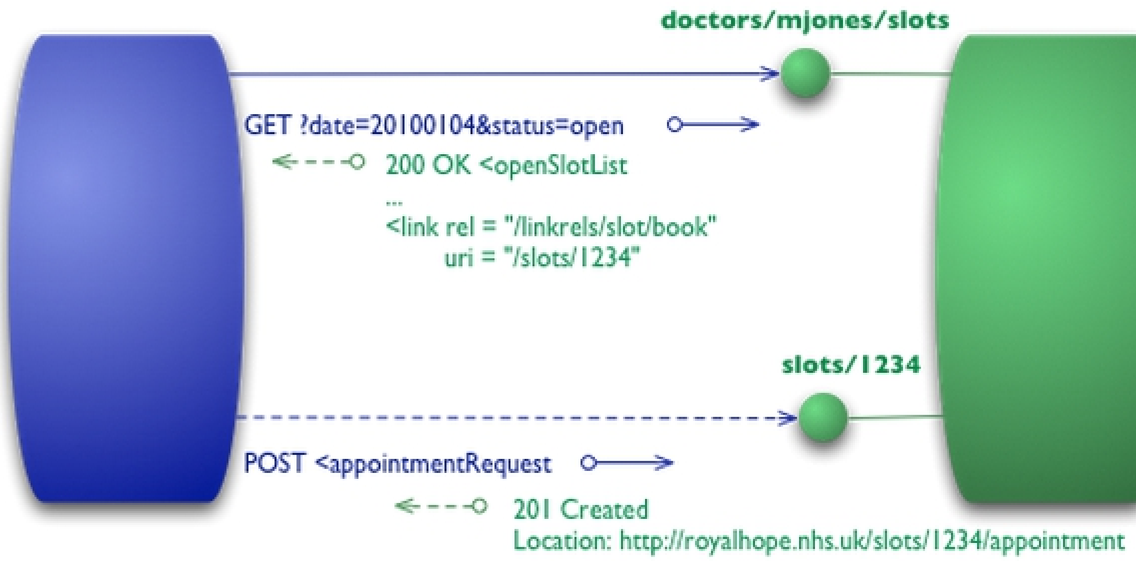
```
HTTP/1.1 409 Conflict
[various headers]

<openSlotList>
  <slot id = "5678" doctor = "mjones" start = "1600" end = "1650"/>
</openSlotList>
```

FIVE

Level 3 : Hypermedia Controls

- HATEOAS
  - Hypermedia as the Engine of Application State
- Svaki response sadrži popis akcija trenutno dopuštenih korisniku
- Prednosti
  - Korisnik zna koje akcije smije izvršiti u datom trenutku
  - Dodatni sloj indirekcije (lakše buduće promjene)
- “Mane”
  - Network overhead
  - Benefit vidljiv tek u long-termu

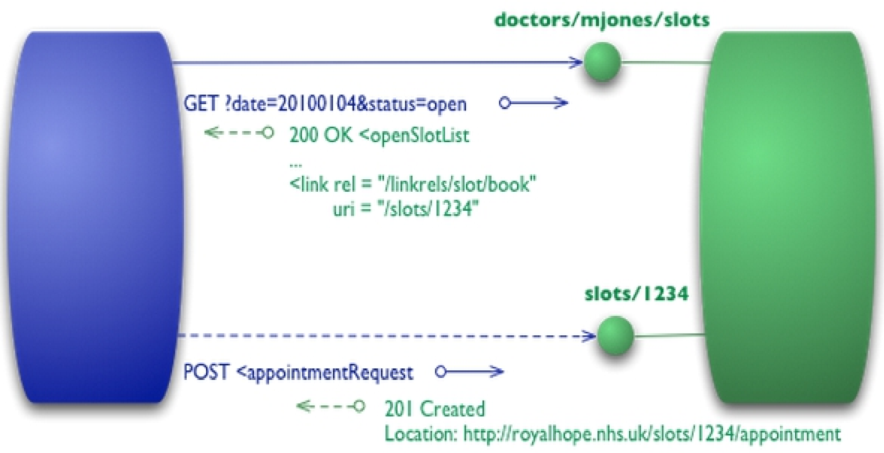
Level 3 : Hypermedia Controls


doctors/mjones/slots
GET ?date=20100104&status=open
200 OK <openSlotList
...
<link rel = "/linkrels/slot/book"
uri = "/slots/1234"
slots/1234
POST <appointmentRequest
201 Created
Location: http://royalhope.nhs.uk/slots/1234/appointment

```
GET /doctors/mjones/slots?date=20100104&status=open HTTP/1.1
Host: royalhope.nhs.uk
```

```
HTTP/1.1 200 OK
[various headers]

<openSlotList>
  <slot id = "1234" doctor = "mjones" start = "1400" end = "1450">
     <link rel = "/linkrels/slot/book"
           uri = "/slots/1234"/>
  </slot>
  <slot id = "5678" doctor = "mjones" start = "1600" end = "1650">
     <link rel = "/linkrels/slot/book"
           uri = "/slots/5678"/>
  </slot>
</openSlotList>
```

Level 3 : Hypermedia Controls


doctors/mjones/slots
GET ?date=20100104&status=open
200 OK <openSlotList
...
<link rel = "/linkrels/slot/book"
uri = "/slots/1234"
slots/1234
POST <appointmentRequest
201 Created
Location: http://royalhope.nhs.uk/slots/1234/appointment

```
POST /slots/1234 HTTP/1.1
[various other headers]

<appointmentRequest>
  <patient id = "jsmith"/>
</appointmentRequest>
```

```
HTTP/1.1 201 Created
Location: http://royalhope.nhs.uk/slots/1234/appointment
[various headers]

<appointment>
  <slot id = "1234" doctor = "mjones" start = "1400" end = "1450"/>
  <patient id = "jsmith"/>
  <link rel = "/linkrels/appointment/cancel"
        uri = "/slots/1234/appointment"/>
  <link rel = "/linkrels/appointment/addTest"
        uri = "/slots/1234/appointment/tests"/>
  <link rel = "self"
        uri = "/slots/1234/appointment"/>
  <link rel = "/linkrels/appointment/changeTime"
        uri = "/doctors/mjones/slots?date=20100104@status=open"/>
  <link rel = "/linkrels/appointment/updateContactInfo"
        uri = "/patients/jsmith/contactInfo"/>
  <link rel = "/linkrels/help"
        uri = "/help/appointment"/>
</appointment>
```

Drugi pokušaj...

HOW RESTFUL IS MY REST ?

HVALA NA PAŽNJI !

PITANJA ?