Prinzipien von REST

Technology
Arts Sciences
TH Köln

Representational State Transfer (REST)

1. Stateless (Session State on Client)
2. Ressourcen mit eindeutiger ID (URI)
3. Verwendung von Standard-Methoden
4. Caching (cacheable)
5. Verschiedene Repräsentationen
6. Applikationszustand via Links
   (Hypermedia As The Engine Of Application State, HATEOAS)

Level 3: **Hypermedia Controls**

“gutes REST- API”

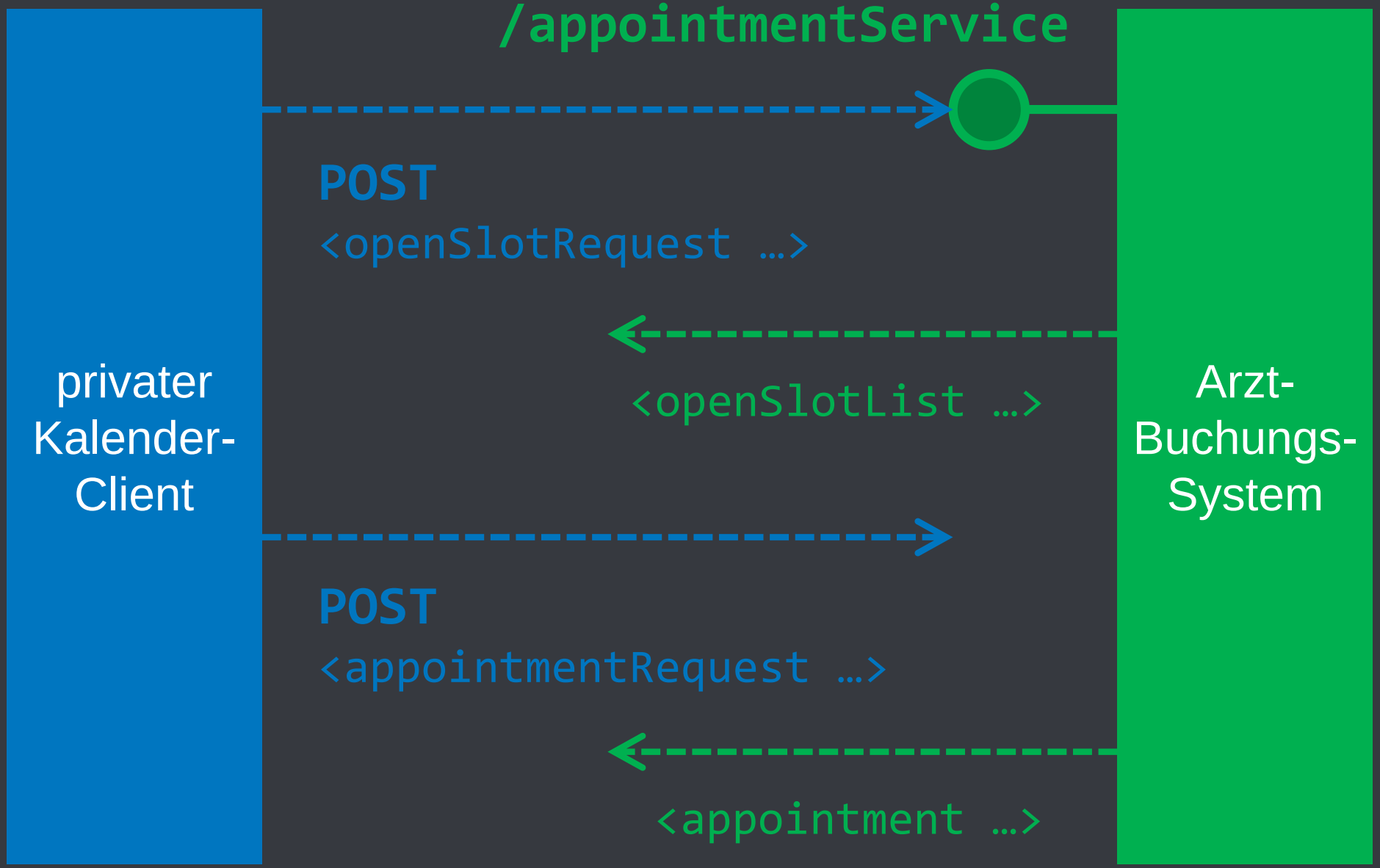
Level 2: **HTTP Verbs**

Level 1: **Resources**

Level 0: **The Swamp of POX**

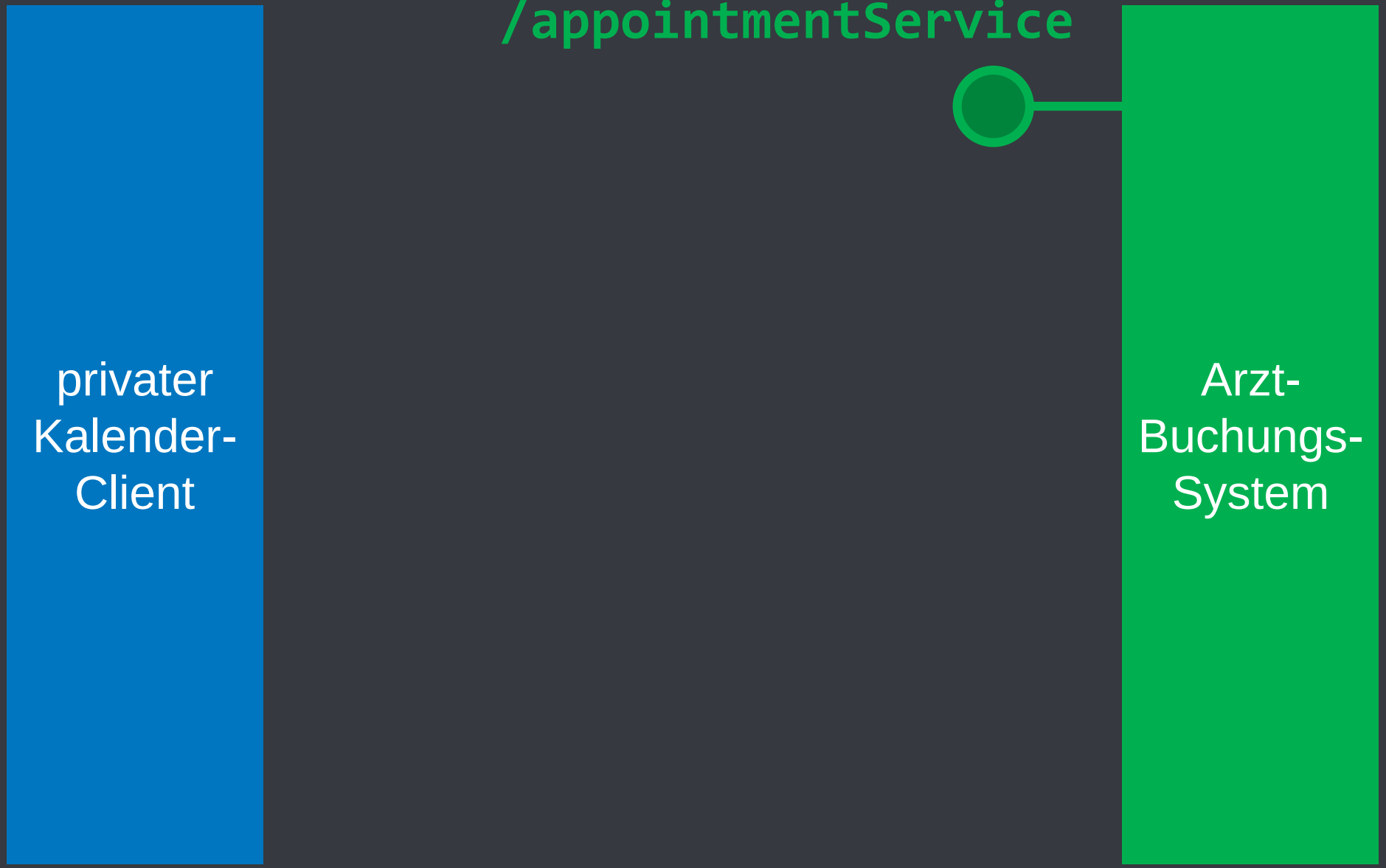
https://martinfowler.com/articles/richardsonMaturityModel.html

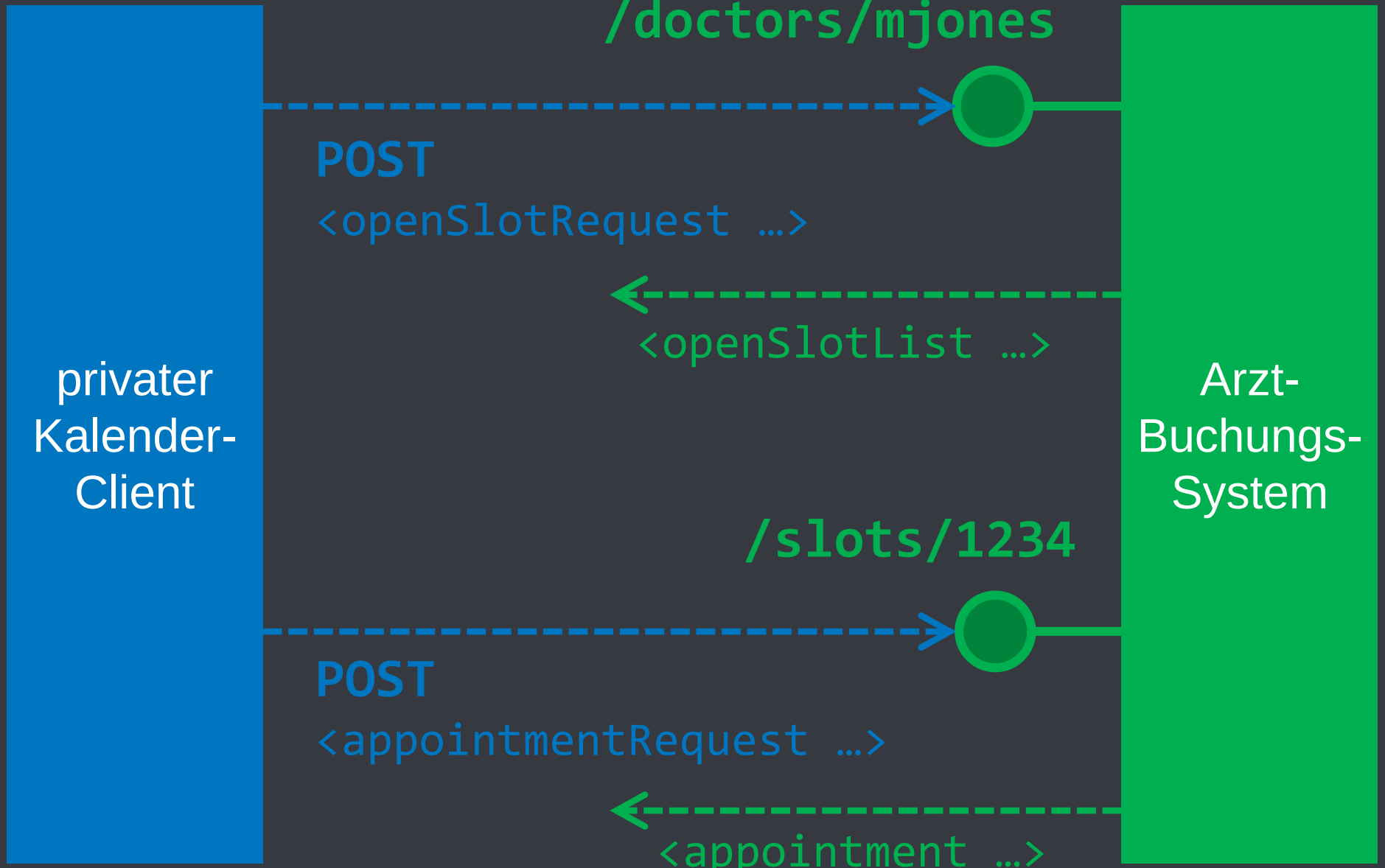
Level 0: **The Swamp of POX**

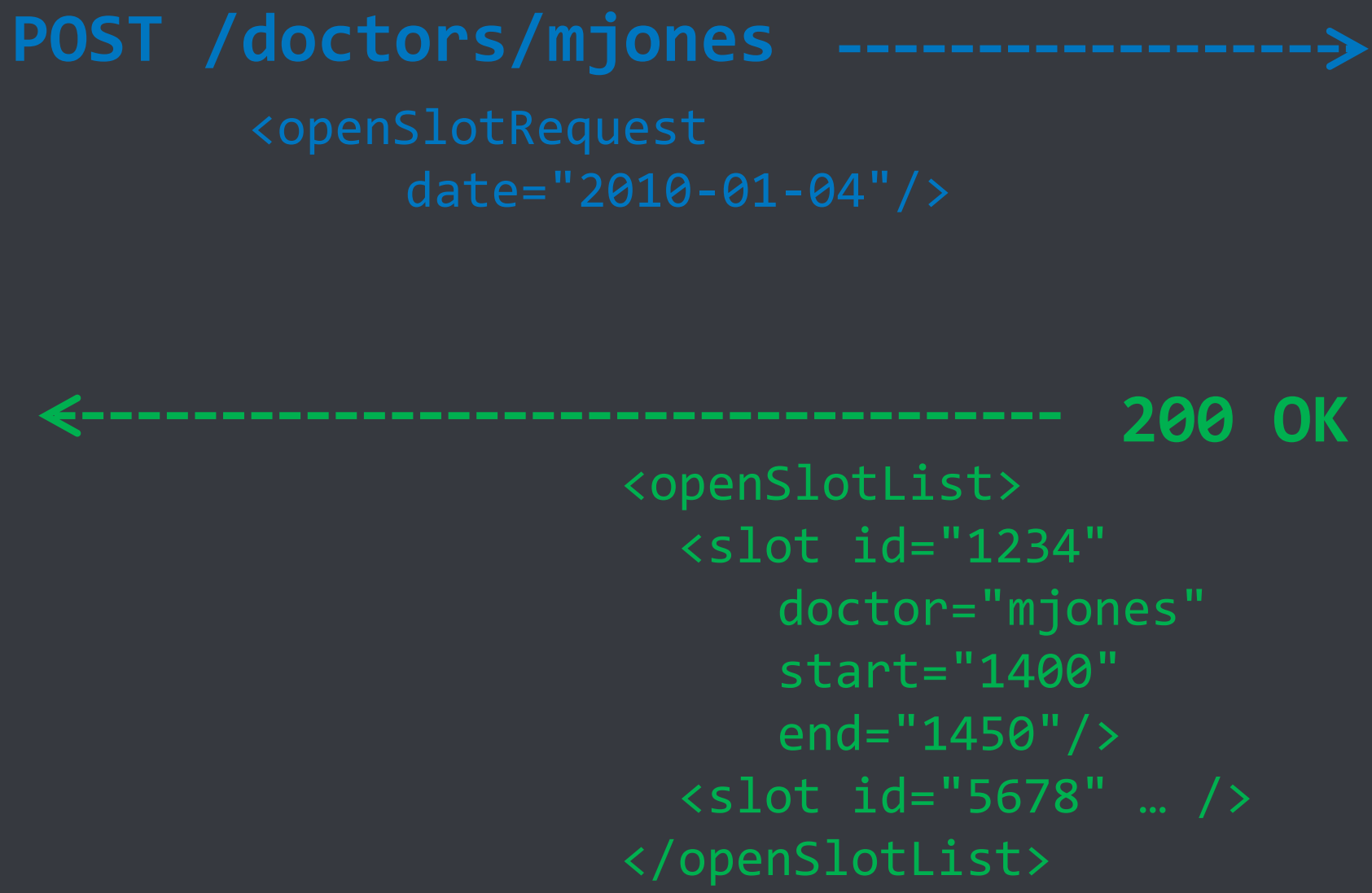

/appointmentService
POST
<openSlotRequest …>
<openSlotList …>
privater Kalender-Client
Arzt-Buchungs-System
POST
<appointmentRequest …>
<appointment …>

	0	1	2	3
Stateless	✓			
Ressourcen mit URI	✗			
Standard-Methoden	✗			
Cacheable	✗			
Repräsentationen	✗			
HATEOAS	✗			

Level 1: **Resources**


/appointmentService
privater Kalender-Client
Arzt-Buchungs-System


/doctors/mjones
POST
<openSlotRequest …>
<openSlotList …>
privater Kalender-Client
Arzt-Buchungs-System
/slots/1234
POST
<appointmentRequest …>
<appointment …>


POST /doctors/mjones
<openSlotRequest
date="2010-01-04"/>
200 OK
<openSlotList>
<slot id="1234"
doctor="mjones"
start="1400"
end="1450"/>
<slot id="5678" … />
</openSlotList>

POST /slots/1234 ---------->

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

<---------- **201 CREATED**

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

	0	1	2	3
Stateless	✓	✓		
Ressourcen mit URI	✗	✓		
Standard-Methoden	✗	✗		
Cacheable	✗	✗		
Repräsentationen	✗	✗		
HATEOAS	✗	✗		

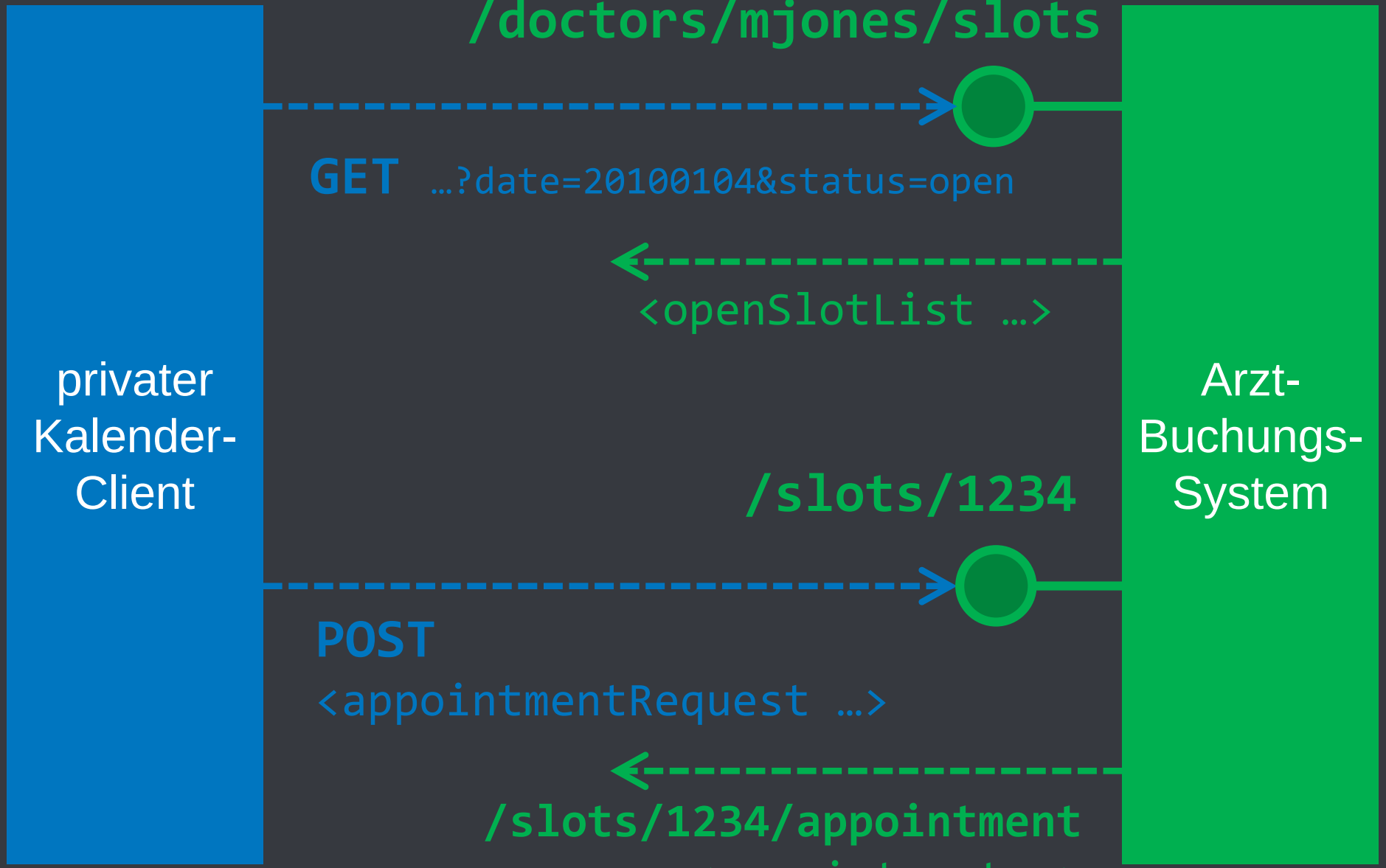
“gutes
REST- API”

Level 2: HTTP Verbs

Eigenschaften der HTTP Verben

- sicher
- idempotent
- identifizierbare Ressource
- Cache-fähig
- sichtbare Semantik

	sicher	idem-potent	identifi-zier-bare Res-source	Cache-fähig	sicht-bare Seman-tik
GET / HEAD	✓	✓	✓	✓	✓
OPTIONS	✓	✓	✓	✗	✓
PUT / PATCH	✗	✓	✓	✗	✓
DELETE	✗	✓	✓	✗	✓
POST	✗	✗	✗	✗	✗


/doctors/mjones/slots
GET …?date=20100104&status=open
<openSlotList …>
privater Kalender-Client
Arzt-Buchungs-System
/slots/1234
POST
<appointmentRequest …>
/slots/1234/appointment
<appointment …>

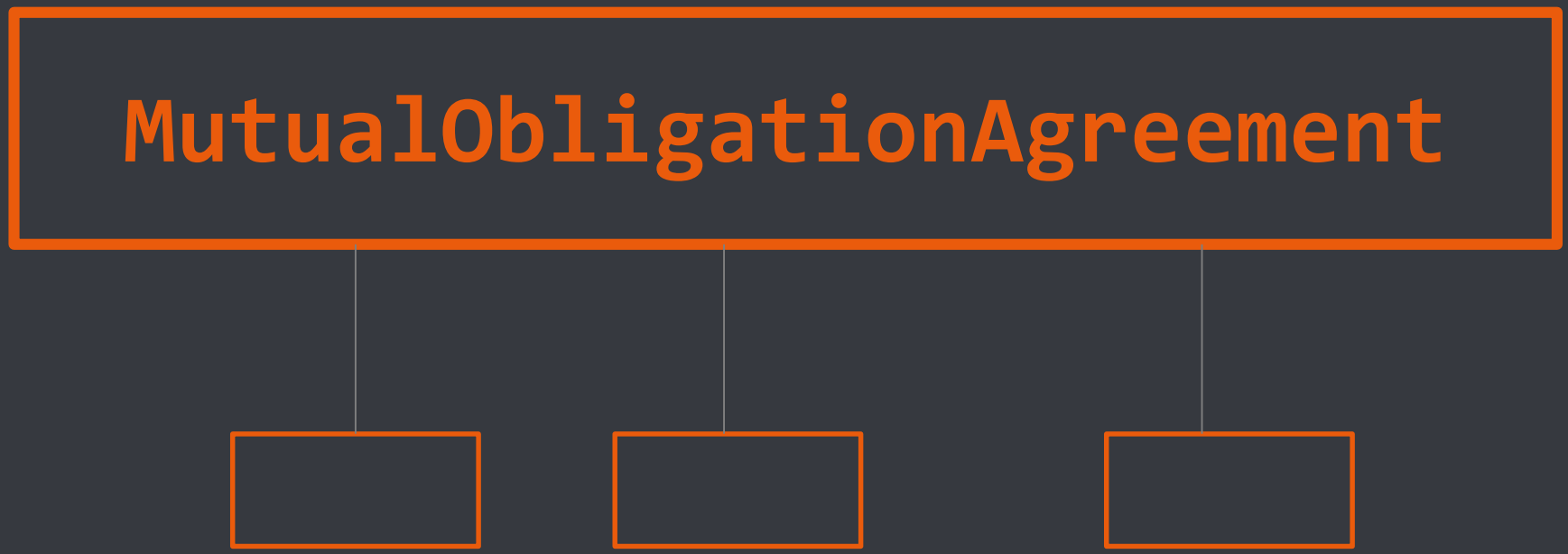
	0	1	2	3
Stateless	✓	✓	✓	✓
Ressourcen mit URI	✗	✓	✓	✓
Standard-Methoden	✗	✗	✓	✓
Cacheable	✗	✗	✓	✓
Repräsentationen	✗	✗	✓	✓
HATEOAS	✗	✗	✗	✓

Kein Dr. Jones?

<---------- 404 NOT FOUND

Termin zwischenzeitlich vergeben?

<---------- **409 CONFLICT**


MutualObligationAgreement

☹

/mutual_obligation_agreement/…

☺

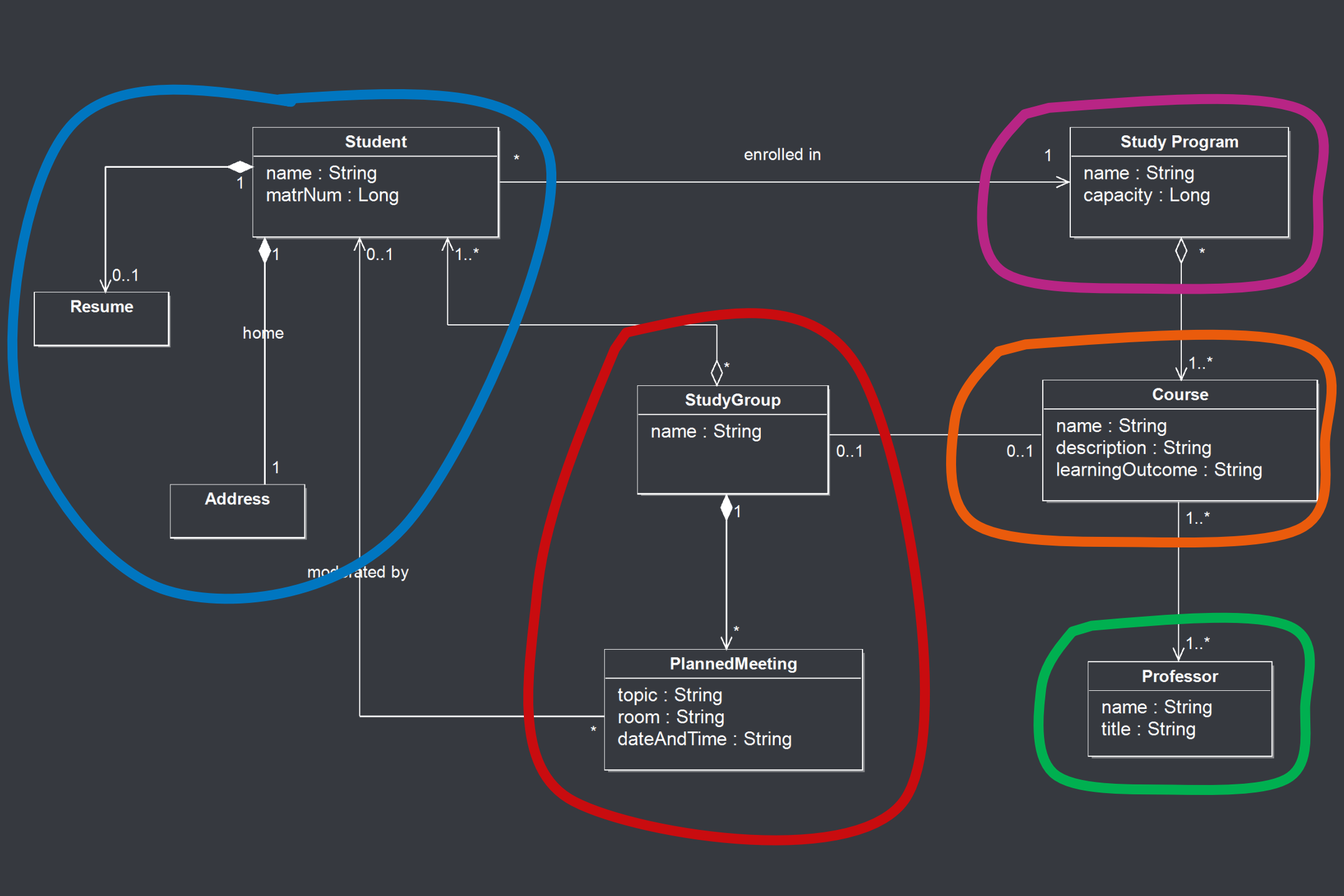
/mutualObligationAgreement/…

/mutual-obligation-agreement/…

Endpoint

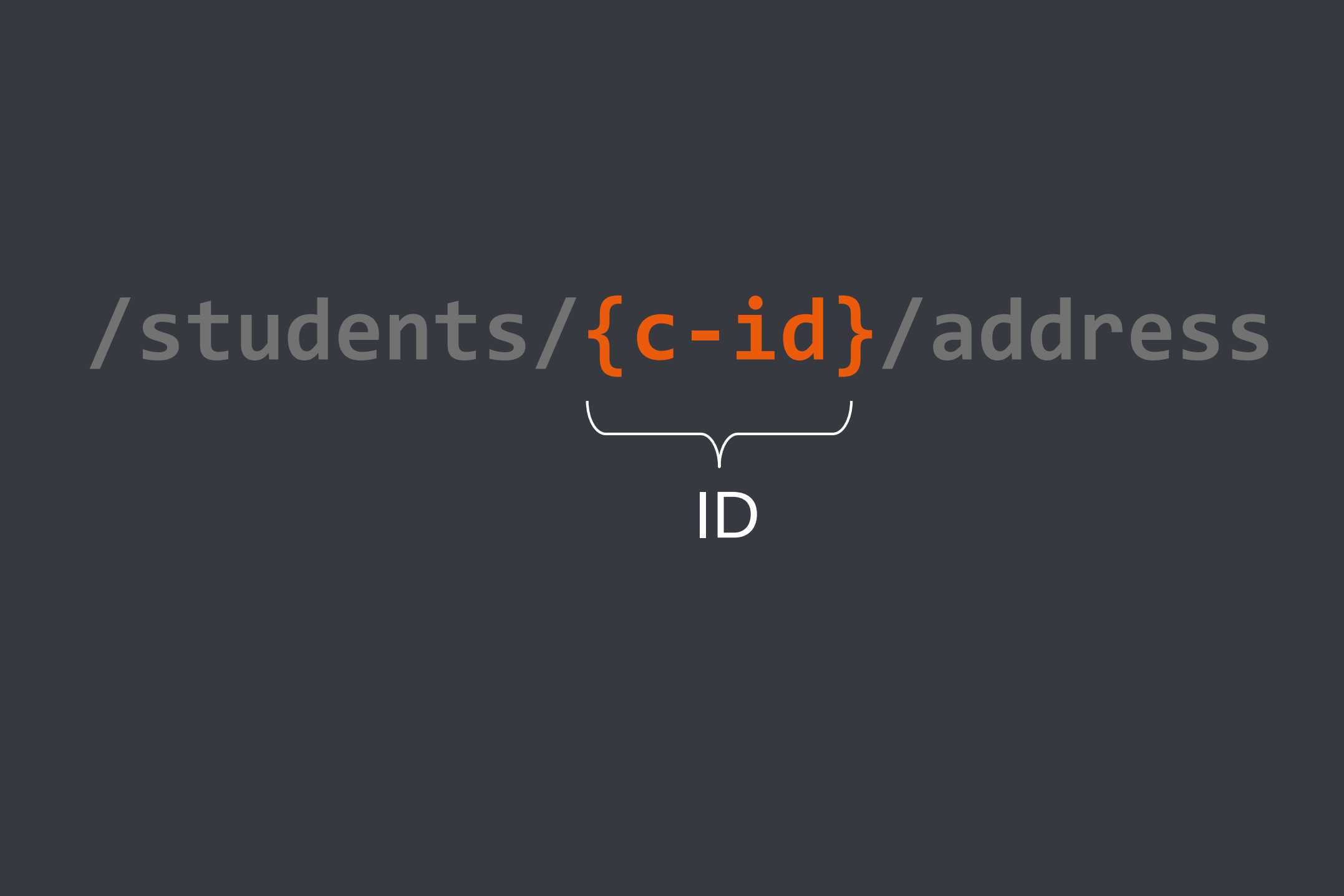
=

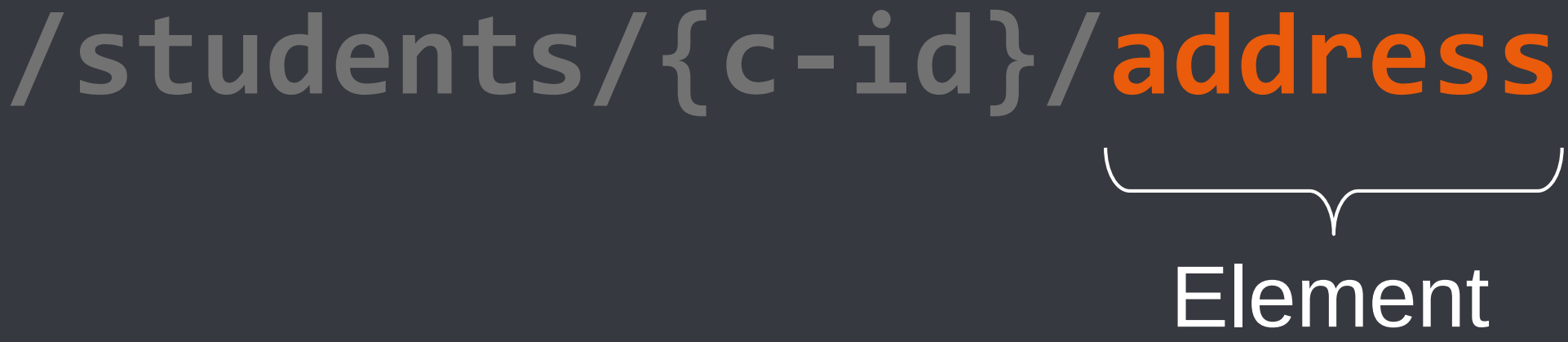
Aggregate


Student
name : String
matrNum : Long
Study Program
name : String
capacity : Long
enrolled in
Resume
Address
home
StudyGroup
name : String
Course
name : String
description : String
learningOutcome : String
PlannedMeeting
topic : String
room : String
dateAndTime : String
Professor
name : String
title : String
moderated by
1
*
0..1
1..*

/students/{c-id}/address


/students/{c-id}/address
Collection


/students/{c-id}/address
ID


/students/{c-id}/address
Element

- HTTP-Verben werden auf das **letzte (rechteste) Element** des URI-Pfads angewendet
- Sie wirken **bezüglich des jeweiligen Parent**

- `GET /studyPrograms`
- `GET /studyPrograms/{s-id}`
- `GET /studyPrograms/{s-id}/courses`

- `DELETE /studyPrograms`
- `DELETE /studyPrograms/{s-id}`
- `DELETE /studyPrograms/{s-id}/courses`

- `POST /studyPrograms`
- `PUT /studyPrograms/{s-id}/courses/{c-id}`
- `PATCH /studyPrograms/{s-id}`

```
GET /mypath?att1=value1&att2=value2
```

- Searching
- Paging
- Filtering
- Sorting

- `GET /students?`
  `matrNumFrom=10000000&`
  `matrNumUntil=39999999`

www.archi-lab.io