

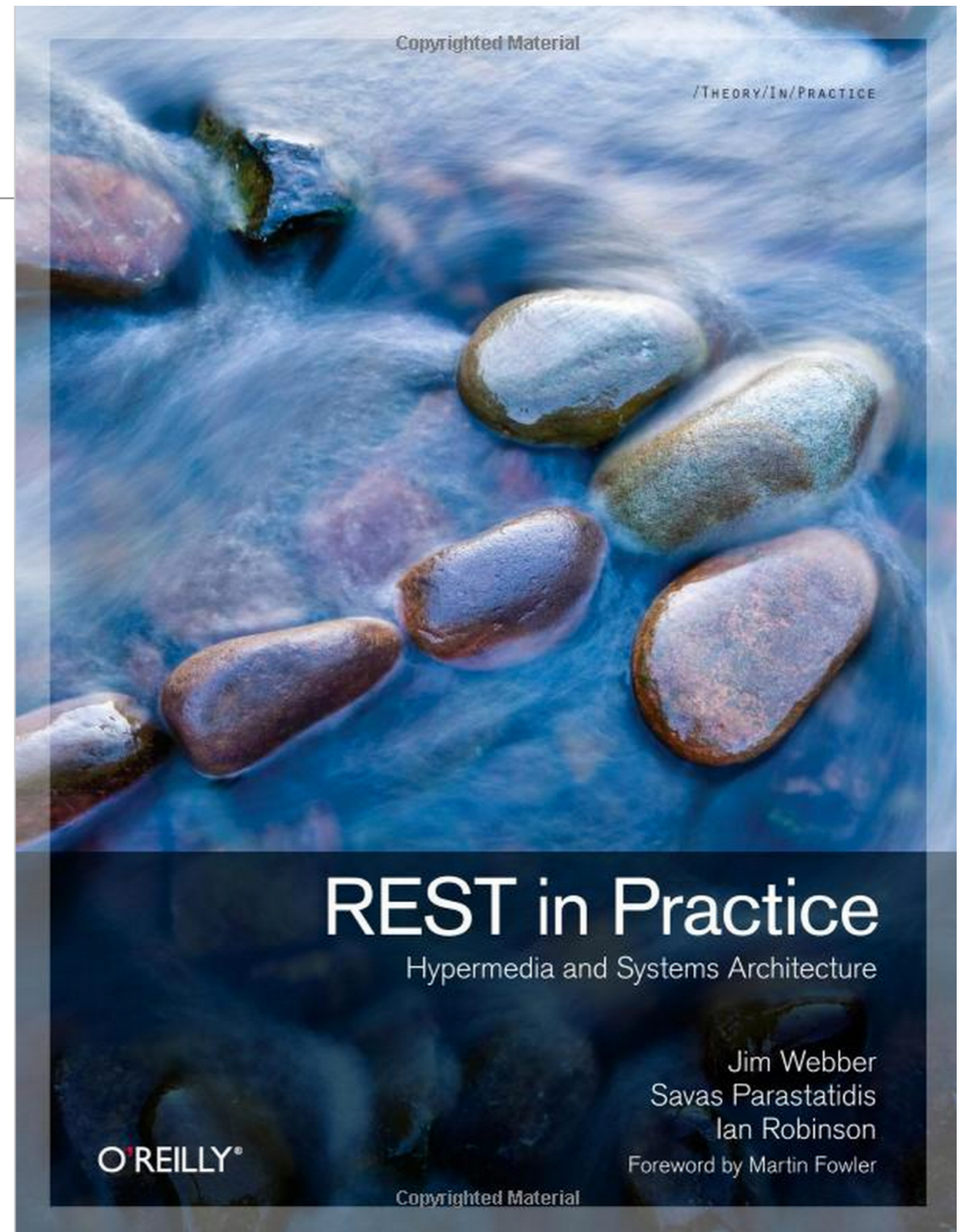
REST

Examples - REST

- Twitter API
- Google Maps
- Twillio
- Github
- Foursquare
- blogger.com
- REST is an “Architectural Style” - enumerating an approach to building distributed systems.
- It embodies an approach that aims to maximize the infrastructure of http infrastructure deployed in the public internet, enabling secure, scalable distributed systems that do not require expensive, complex alternative infrastructure.

REST

Representational State Transfer (REST) is an architectural style that abstracts the architectural elements within a distributed [hypermedia](#) system. [\[1\]](#) REST ignores the details of component implementation and protocol syntax in order to focus on the roles of components, the constraints upon their interaction with other components, and their interpretation of significant data elements. [\[2\]](#) REST has emerged as a predominant [web API](#) design model



REST: The Web Used Correctly

- A system or application architecture
- ... that uses HTTP, URI and other Web standards “correctly”
- ... is “on” the Web, not tunnelled through it ... also called ““RESTful HTTP””

(see http://www.slideshare.net/deimos/stefan-tilkov-pragmatic-intro-to-rest?qid=bf0300ad-6e68-4faa-bac7-0f77424ec093&v=qf1&b=&from_search=1)

Rest Principles

- 1: Give Everything and ID
- 2: Link Things Together
- 3: Use Standard HTTP Methods
- 4: Allow for Multiple Representations
- 5: Communicate Statelessly

1: Give Every Thing and ID

- <http://example.com/customers/1234>
- <http://example.com/orders/2007/10/776654>
- <http://example.com/products/4554>
- <http://example.com/processes/sal-increase-234>

2: Link Things Together

```
<order self='http://example.com/orders/1234'>  
  <amount>23</amount>  
  <product ref='http://example.com/products/4554' />  
  <customer ref='http://example.com/customers/1234' />  
</order>
```

3: Use Standard HTTP Methods

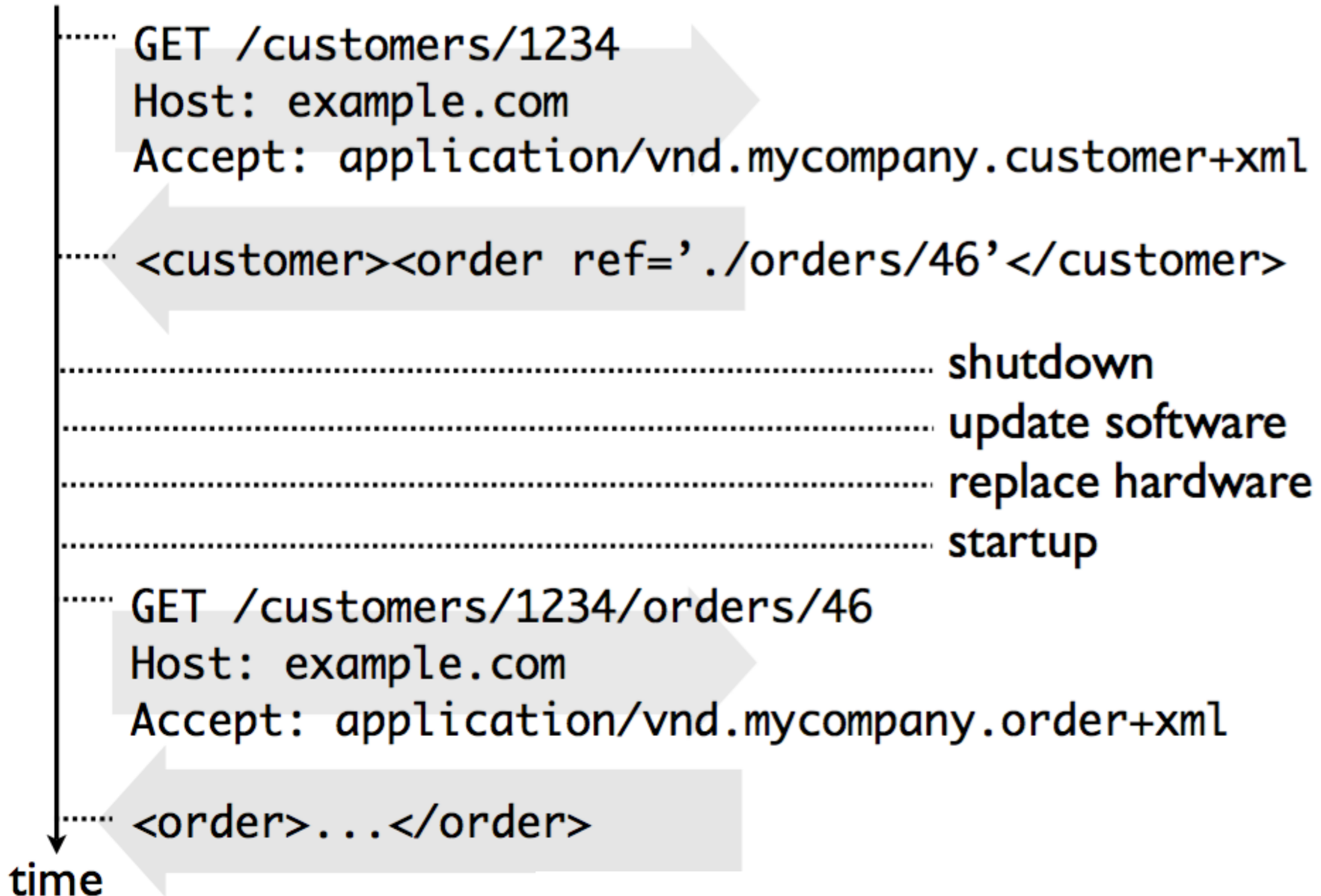
GET	retrieve information, possibly cached
PUT	Update or create with known ID
POST	Create or append sub-resource
DELETE	(Logically) remove

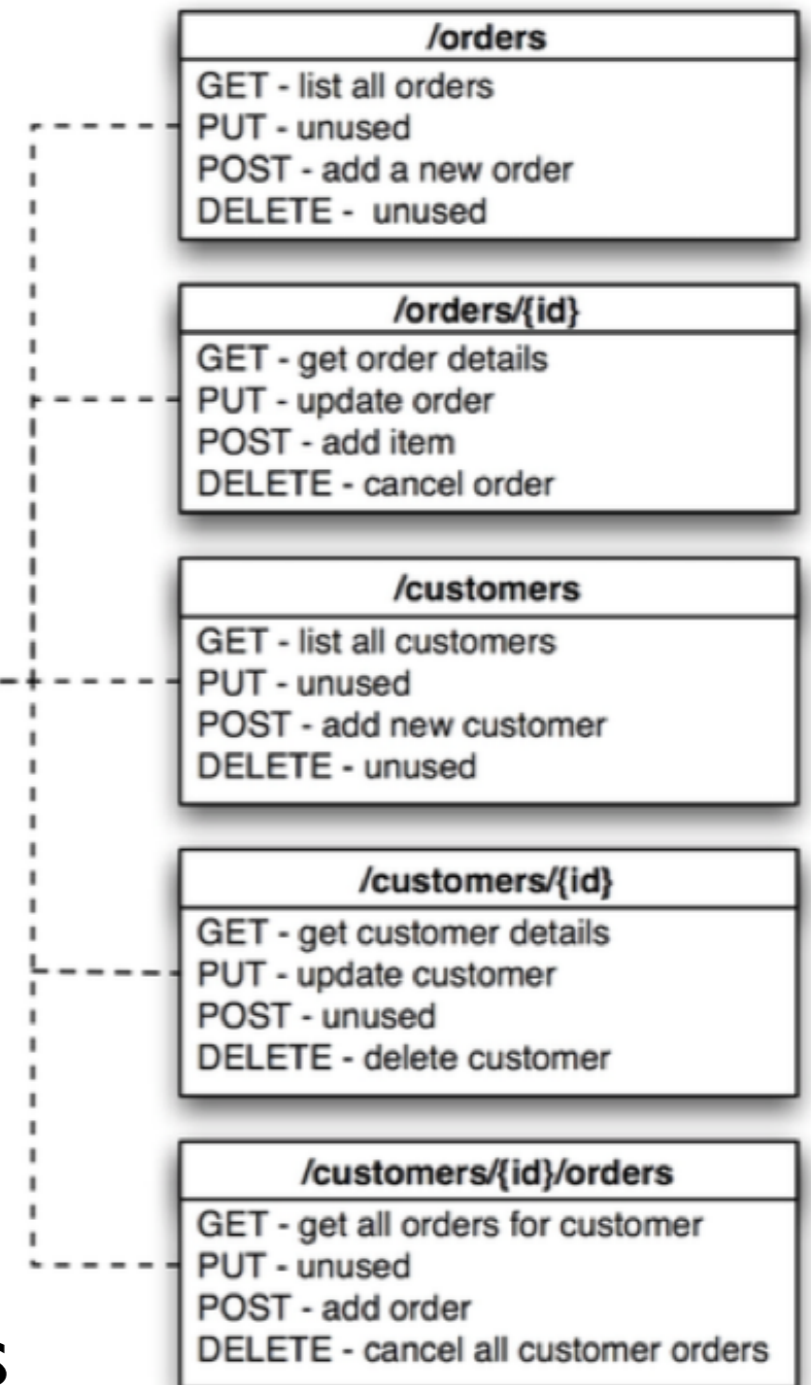
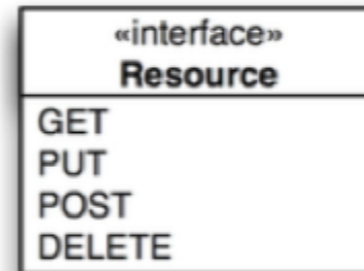
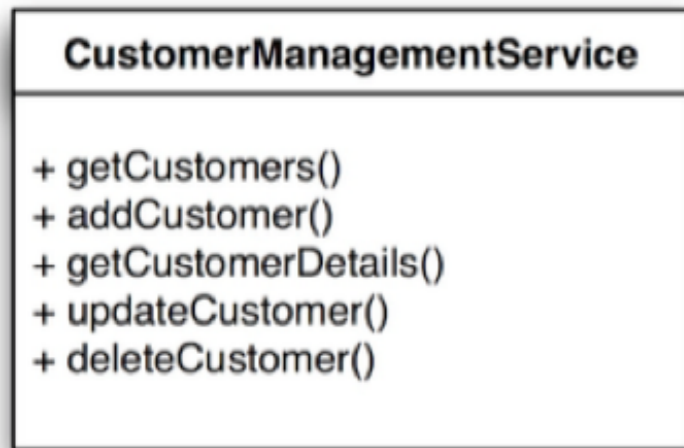
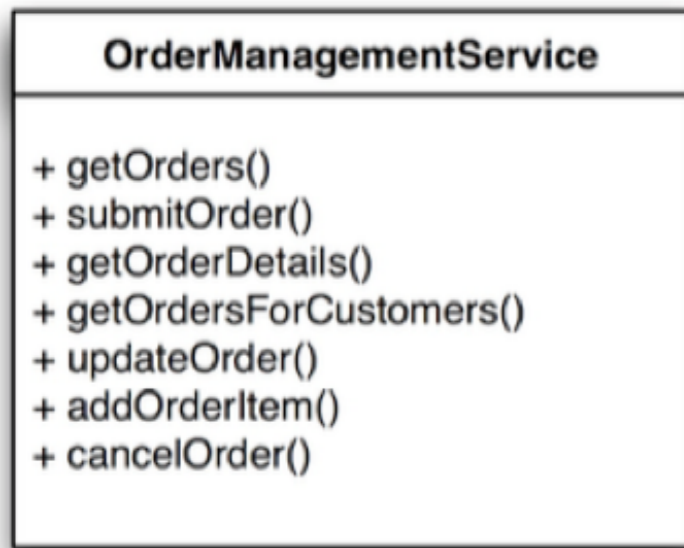
4: Allow for Multiple Representations

```
GET /donors/1234
Host: example.com
Accept: application/json
{
  "firstName" : "fred",
  "lastName"  : "simpson",
  "email"     : "fred@simpson.com",
  "password"  : "secret"
}
```

```
GET /donors/1234
Host: example.com
Accept: application/xml
<donor>
  <firstName> "fred" </firstName>
  <lastName>  "simpson" </lastName>
  <email>    "fred@simpson.com" </email>
  <password> "secret" </password>
</donor>
```

5: Communicate Statelessly





Identify resources & design URIs

Select format (json)

Identify method semantics

Select response codes