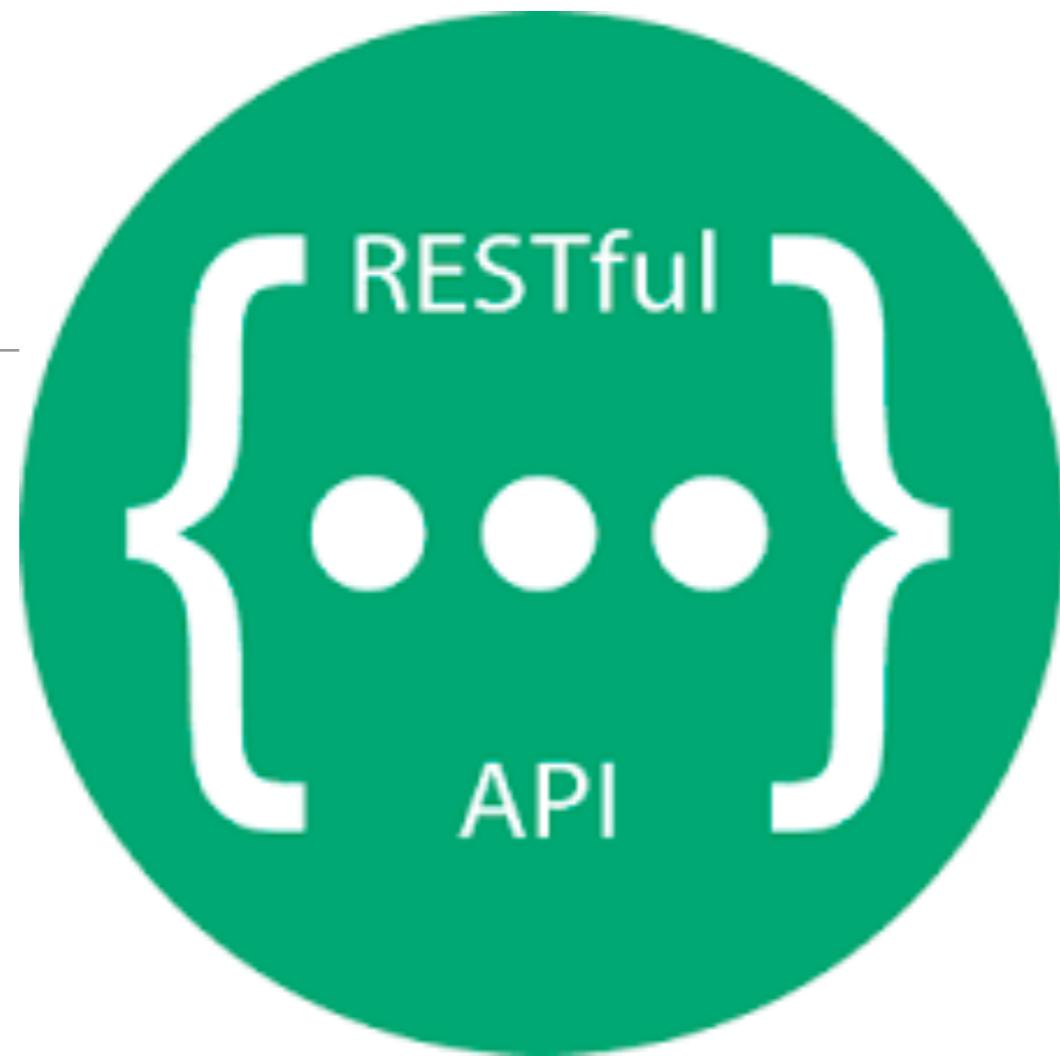


REST : Representation State Transfer



Examples - REST

RESTful API

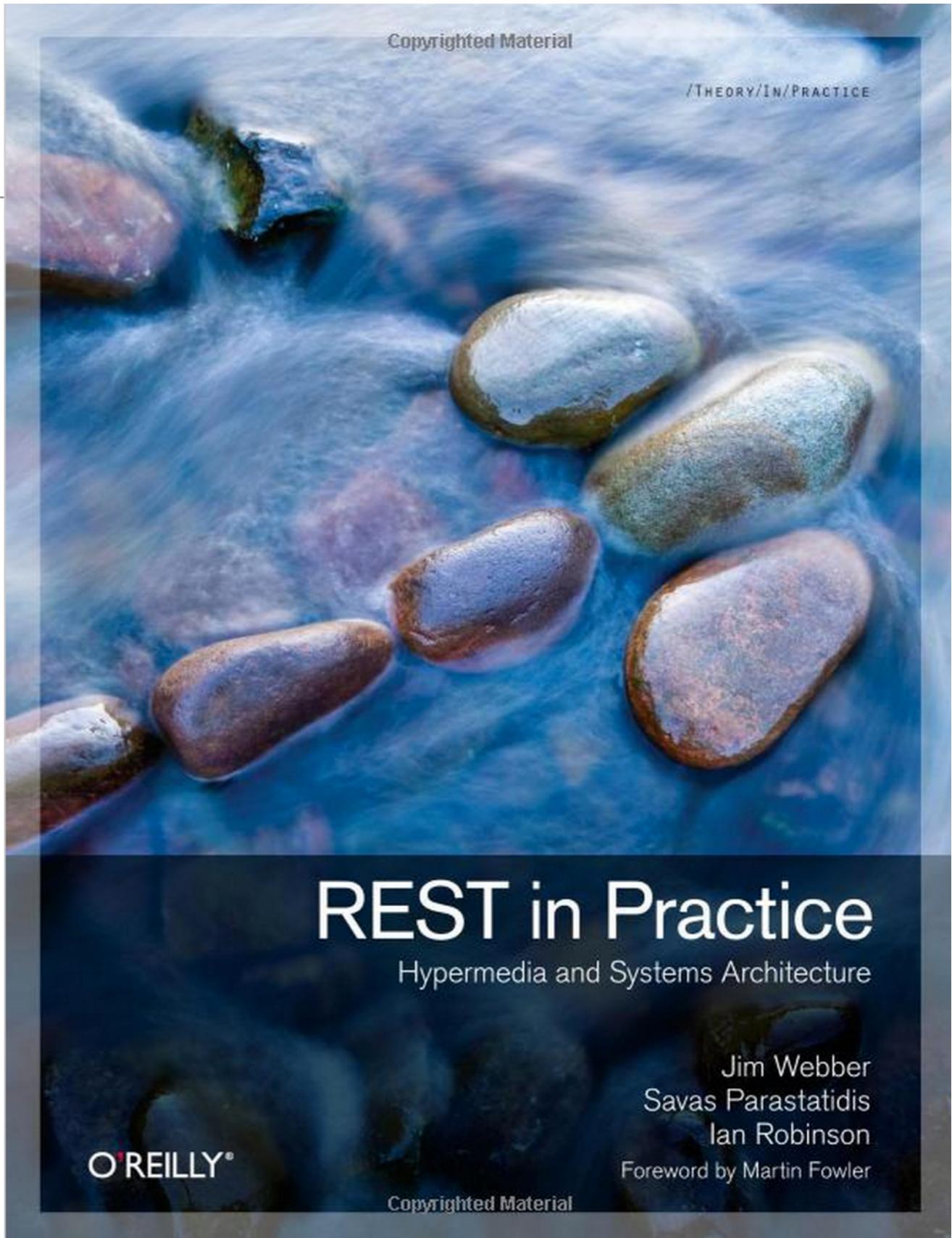
GET PUT POST DELETE

- Twitter API
 - Google Maps
 - Twilio
 - 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

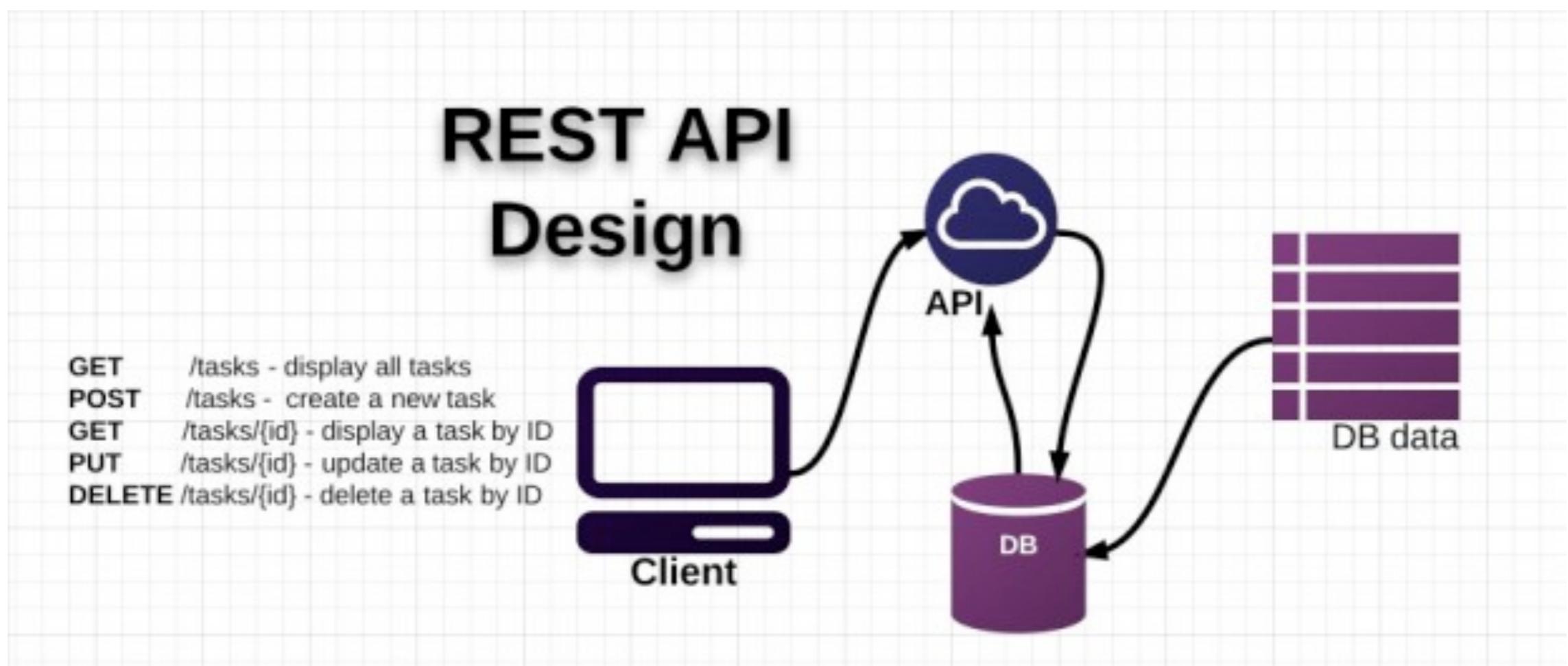
Representational State Transfer

(**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””



Rest Principles

- 1: Give Everything an 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>



504a4e0-8497-4f83-8eaa-000000000000
id:>c225ae5c-540f-4a48-8867-809b395...
1:uuid:>1e1bd59-3dd6-4f96-9166-e6095e7231...
urn:uuid:>74efb6ba-a52a-46c0-a16b-03860d356882...
"urn:uuid:>4d0ecbdb-4cba-4047-8351-29283adf67c7<...
"urn:uuid:>20f19a35-401b-45a6-a54e-084122a4cf80<...
'urn:uuid:>3a17efd0-adfe-4899-9d4c-5b8ac591645b<...
n:uuid:>02e42ef0-7be7-4c60-b8d9-790b79fa38?...
id:>6c8b3cb4-c0fe-4af8-ac63-cb2a9060...
~5f046a-e7aa-44e8-9712-2...

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

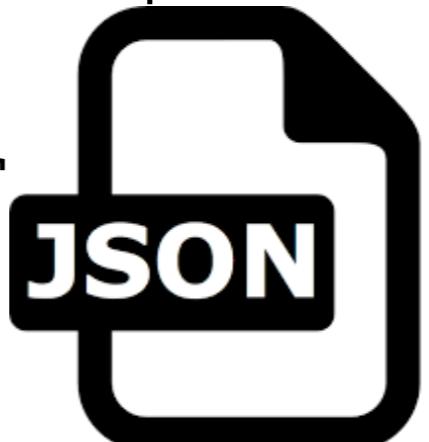
{

 "firstName" : "fred",

 "lastName" : "simpson",

 "email" : "fred@simpson.com",

 "password" : "secre



GET /donors/1234

Host: example.com

Accept: application/xml

<donor>

 <firstName> "fred" </firstName>

 <lastName> "simpson" </lastName>

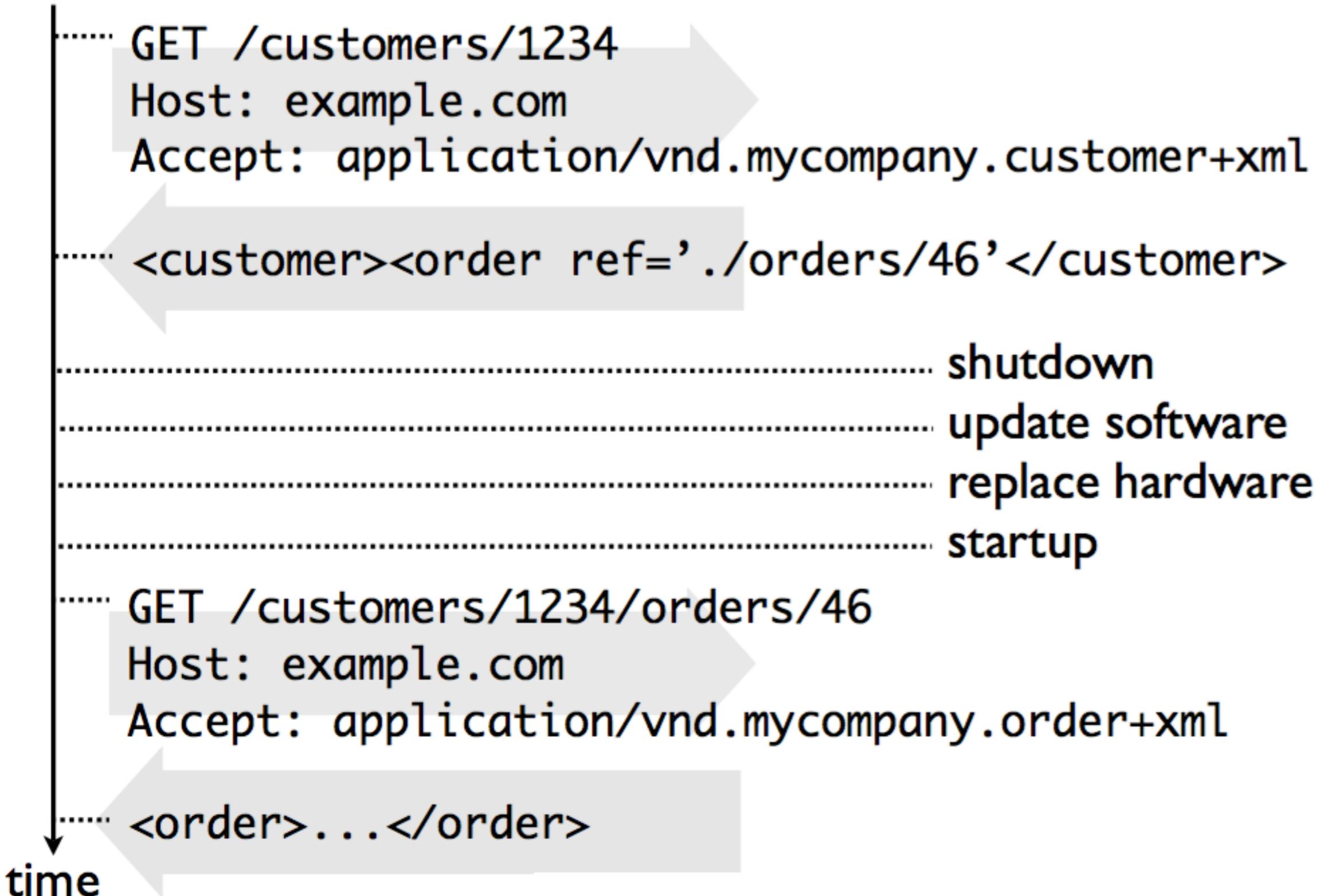
 <email> "fred@simpson.com" </email>

 <password> "secret"



</donor>

5: Communicate Stateless



OrderManagementService
+ getOrders()
+ submitOrder()
+ getOrderDetails()
+ getOrdersForCustomers()
+ updateOrder()
+ addOrderItem()
+ cancelOrder()

CustomerManagementService
+ getCustomers()
+ addCustomer()
+ getCustomerDetails()
+ updateCustomer()
+ deleteCustomer()



«interface» Resource
GET
PUT
POST
DELETE

/orders
GET - list all orders
PUT - unused
POST - add a new order
DELETE - unused
/orders/{id}
GET - get order details
PUT - update order
POST - add item
DELETE - cancel order
/customers
GET - list all customers
PUT - unused
POST - add new customer
DELETE - unused
/customers/{id}
GET - get customer details
PUT - update customer
POST - unused
DELETE - delete customer
/customers/{id}/orders
GET - get all orders for customer
PUT - unused
POST - add order
DELETE - cancel all customer orders

Rest Endpoints Verbs

- Comparing database (sql) and HTTP Verbs

<u>SQL</u>	<u>REST</u>
SELECT	GET
INSERT	POST
UPDATE	PUT
DELETE	DELETE

Action varies with HTTP Method

URI	HTTP METHOD	ACTION PERFORMED
/status/	GET	Get all status
/status/3	GET	Get status with id 3
/status/	POST	Add a new status
/status/4	PUT	Edit status with id 4
/status/4	DELETE	Delete status with id 4

HTTP Response Codes

HTTP Status Codes	Informational
200	OK
201	Resource created
204	No content
400	Bad Request
401	Unauthorised
404	Not found
405	Method Not allowed
500	Internal Server Error