Aprameyah
Technologies Pvt. Ltd.

ReST

Representational State Transfer

Aprameyah
Technologies Pvt. Ltd.

Web As A Resource

- Roy Fielding - co-founder - apache http server project.
    - His doctoral thesis describes REST as an architectural style
    - Why is the Web so prevalent and ubiquitous?
    - What makes the Web scale?
    - How can I apply the architecture of the Web to my own applications?

Aprameyah
Technologies Pvt. Ltd.

What's ReST

- It relies on a stateless, client-server, cacheable communications protocol -- and in virtually all cases, the HTTP protocol is used.

- REST is a lightweight alternative to mechanisms like RPC (Remote Procedure Calls) and Web Services (SOAP, WSDL, et al.).

- REST is not a "standard". There will never be a W3C recommendation for REST,

Aprameyah
Technologies Pvt. Ltd.

Examples

- Google search API
- Google Maps API
- Facebook People Search API

Aprameyah
Technologies Pvt. Ltd.

Addressable Resources

- Every object and resource in your system is reachable through a unique identifier.
- Does not mean we expose the underlying data model as is.
    - Many times we create API models that are a combination of data models

Aprameyah
Technologies Pvt. Ltd.

How is biz logic exposed

- RPC Style interface operation example:
  - val = calcSum(a,b)
- REST style interface treating the sum as a resource
  - http://host/sum?op1=a&op2=b
  - Cacheable resource
  - Idempotent

Aprameyah
Technologies Pvt. Ltd.

How is biz logic exposed

OPERATION	RPC (OPERATION)	REST (RESOURCE)
Signup	POST /signup	POST /persons
Resign	POST /resign	DELETE /persons/1234
Read a person	GET /readPerson?personid=1234	GET /persons/1234
Read a person's items list	GET /readUsersItemsList?userid=1234	GET /persons/1234/items
Add an item to a person's list	POST /addItemToUsersItemsList	POST /persons /1234/items
Update an item	POST /modifyItem	PUT /items/456
Delete an item	POST /removeItem?itemId=456	DELETE /items/456

Aprameyah
Technologies Pvt. Ltd.

Sometimes They Dont Fit Well

- Sometimes the concept of resource does not fit well.
- Example: google map direction finding rest api could have been called “findDirection” to indicate that its a method
  - However google makes it a resource and calls it “direction”. A noun is used instead of verb
  - http://maps.googleapis.com/maps/api/directions/json?origin=Jakkur&destination=Hebbal
  - This makes cacheing easy with http semantics
- Think of resources as classes (nouns) and not methods (verbs)

Aprameyah
Technologies Pvt. Ltd.

Verbs and Meanings

VERB	MEANING	IDEMPOTENT	SAFE	CACHEABLE
GET	Reads a resource	Yes	Yes	Yes
POST	Creates a resource or triggers a data-handling process	No	No	Only cacheable if response contains explicit freshness information
PUT	Fully updates (replaces) an existing resource or create a resource	Yes	No	No
PATCH	Partially updates a resources	No	No	Only cacheable if response contains explicit freshness information
DELETE	Deletes a resource	Yes	No	No

Aprameyah
Technologies Pvt. Ltd.

Constrained Interface

- APIs dont have an “action” parameter. The set of operations are pre-defined - GET/POST/PUT etc
- Builds familiarity and semantics
- Improves Interoperability
- Policy based scaling because of predictable behaviour

Aprameyah
Technologies Pvt. Ltd.

Characteristics

- Resources, which are identified by logical URLs. Both state and functionality are represented using resources.

- REST interfaces should return user interface content that is independent of how the user interface will be displayed.

- A web of resources, meaning that a single resource should not be overwhelmingly large and contain too fine-grained details. Whenever relevant, a resource should contain links to additional information

- There is no connection state

- Resources should be cacheable whenever possible

Aprameyah
Technologies Pvt. Ltd.

Guiding Principles Of REST

- Do not use "physical" URLs: "http://www.acme.com/inventory/product003.xml". A logical URL does not imply a physical file: “http://www.acme.com/inventory/product/003".

- Queries should not return an overload of data. If needed, provide a paging mechanism.

- Even though the REST response can be anything, make sure it's well documented

- Rather than letting clients construct URLs for additional actions, include the actual URLs with REST responses

- GET/PUT/DELETE access requests should be idempotent

Aprameyah
Technologies Pvt. Ltd.

Richardson Maturity Model

- Level 0 - SOAP et all - Single verb and single URI
- Level 1 - Resources
- Level 2 - HTTP Verbs and Status Codes
- Level 3 - Hypermedia

Aprameyah
Technologies Pvt. Ltd.

HATEOAS

- Hypermedia as the Engine of Application State
- The principle is that a client interacts with a network application entirely through hypermedia provided dynamically by application servers.
- Links in a document also identify state change
- Older applications usually have a list of precanned services they know exist, and they interact with a central directory server to locate these services on the network

Example of HATEOAS


```
{
   "content": [ {
       "price": 499.00,
       "description": "Apple tablet device",
       "name": "iPad",
       "links": [ {
           "rel": "self",
           "href": "http://localhost:8080/product/1"
       } ],
       "attributes": {
           "connector": "socket"
       }
   }, {
       "price": 49.00,
       "description": "Dock for iPhone/iPad",
       "name": "Dock",
       "links": [ {
           "rel": "self",
           "href": "http://localhost:8080/product/3"
       } ],
       "attributes": {
           "connector": "plug"
       }
   } ],
   "links": [ {
       "rel": "product.search",
       "href": "http://localhost:8080/product/search"
   } ]
}
```

Aprameyah
Technologies Pvt. Ltd.

Link Data

- rel - indicates the type of link: next, previous, edit
- href - the actual location of the resource
- type - the content-type produced by the url

Aprameyah
Technologies Pvt. Ltd.

Pagination With HATEOAS

Clients dont have to construct a set of parameters or URLs for a page.. they just use the urls in current data to reach next set of data

```
<products>
      <link rel="next" href="http://example.com/store/products?
startIndex=5"/>
      <product id="123">
         <name>headphones</name>
         <price>$16.99</price>
      </product>
   ...
   </products>
```

Aprameyah
Technologies Pvt. Ltd.

REST and Scaling

- Modern backends are not consistent
    - CAP Theorem shows that consistent systems do not scale
    - We fall back on eventual consistency instead of instantaneous consistency
- Transactions are not perfect in large scale systems

Making A Purchase

- Something like amazon.com need replicated databases for failover (No concept of backups)
- An order placed on one database instance has to be replicated to all.
    - There is a moment when the data is inconsistent on other db
- Idempotency of REST operations like PUT helps just repeat operation on inconsistent instances

JAX-RS

- JAX-RS is a framework that focuses on applying Java annotations to plain Java objects to create REST APIs.

- It has annotations to bind specific URI patterns and HTTP operations to individual methods of your Java class.

- It has parameter injection annotations so that you can easily pull in information from the HTTP request.

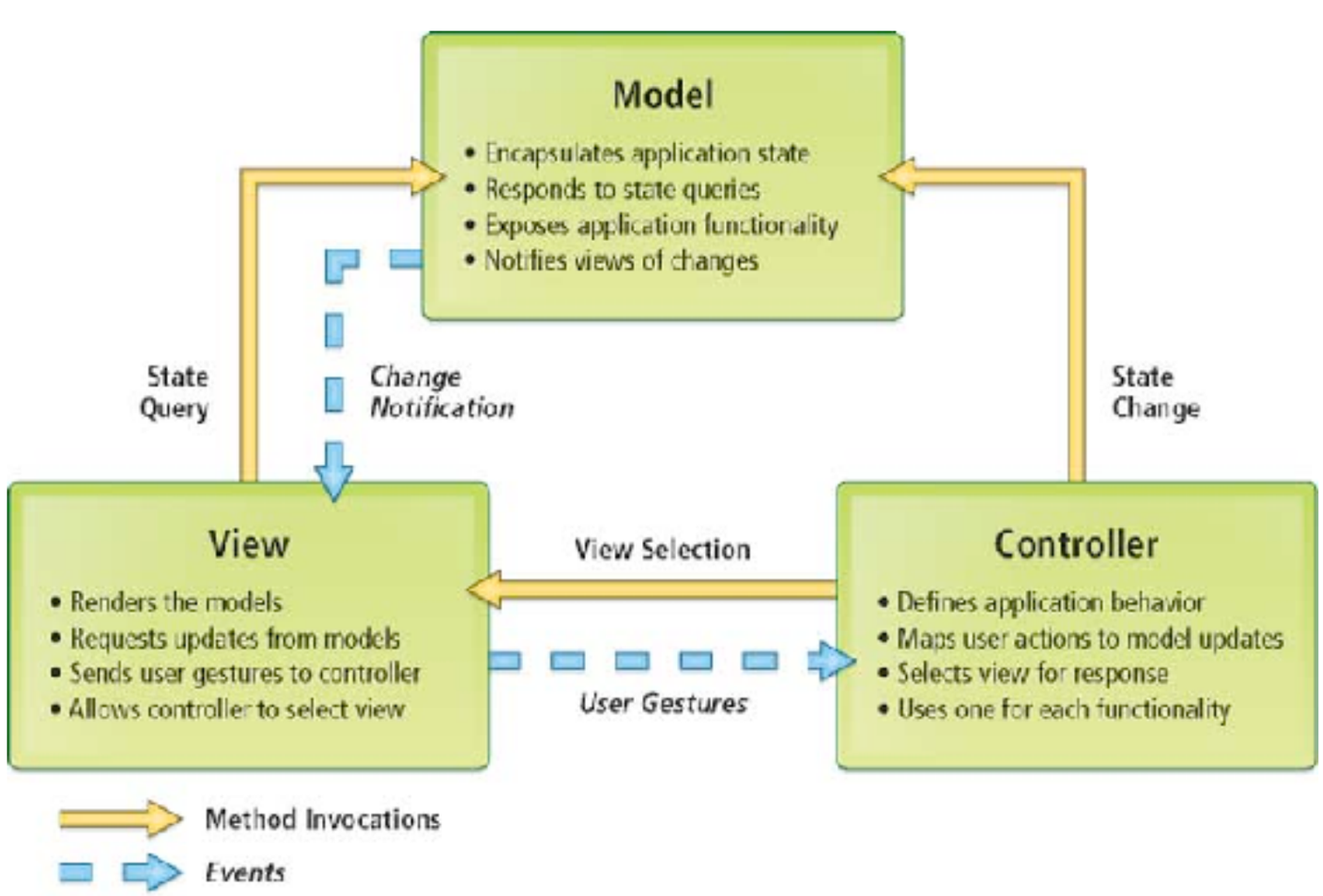
Aprameyah
Technologies Pvt. Ltd.

Spring Rest

- Spring Rest is built on top of Spring MVC
- Parameter and JSON handling is provided by spring and jackson libraries

Spring MVC

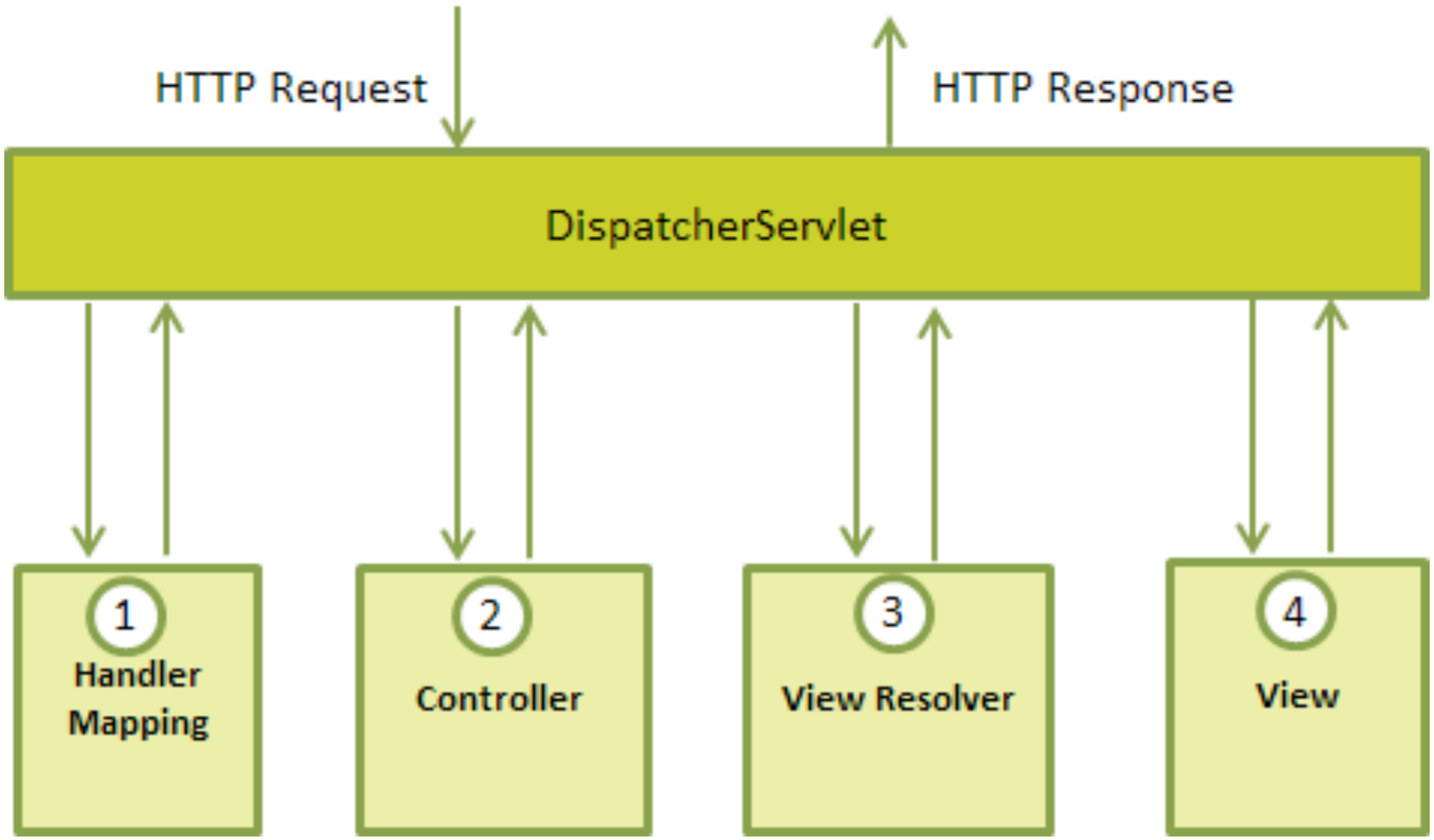
- The Spring web MVC framework provides model-view-controller architecture and ready components that can be used to develop flexible and loosely coupled web applications.



Aprameyah Technologies Pvt. Ltd.

flikbuk

Spring MVC Flow



Aprameyah Technologies Pvt. Ltd.

flikbuk

Hello World

- Create a maven app with webapp archetype
- Setup dependencies on Spring MVC and related libraries
- Web.xml

```
<servlet>
      <servlet-name>action</servlet-name>
      <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
      <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
      <servlet-name>action</servlet-name>
      <url-pattern>/action/*</url-pattern>
</servlet-mapping>
```

- WEB-INF/action-servlet.xml

```
<context:component-scan base-package="com.mydomain"/>
```

- Controller class

```
@Controller
public class HelloWorldController {

    @RequestMapping("/helloWorld")
    public String helloWorld(Model model) {
        model.addAttribute("message", "Hello World!");
        return "/helloWorld";
    }
}
```

- helloWorld.jsp - put this in WEB-INF\Views folder (this folder needs to be created)

```
<%@ page isELIgnored="false"%>
<h2>${message}</h2>
```

Aprameyah
Technologies Pvt. Ltd.

Passing Form Data

- Declare controller that accepts form data as a bean

```
@Controller
public class UserController {

    @Autowired
    UserManager userManager;

    @RequestMapping("/addUser")
    public String addUser(User u, Model model) throws Exception {
      userManager.addUser(u);
        model.addAttribute("message", "Added User!");
        return "/helloWorld.jsp";
    }
}
```

- Submit form to this servlet

Aprameyah
Technologies Pvt. Ltd.

Lets Build REST services

- Create a maven app with webapp archetype
- Setup dependencies on Spring MVC and related libraries
- Add this to action-servlet.xml

```
<mvc:annotation-driven />
```

- Create a root resource class

```
@RestController
@RequestMapping("/users")
public class UserResource {

    @RequestMapping(method=RequestMethod.GET, value="/",
            produces=MediaType.APPLICATION_JSON_VALUE)
    public List<User> getUsers() {
        //Load a list of User objects using Hibernate and return
    }
}
```

Aprameyah
Technologies Pvt. Ltd.

Using Javascript Client

- Include JQuery into a html

```
$.get("/url",function(data,status){

})

$.post(“/url",data,function(data,status){

})
```

Aprameyah
Technologies Pvt. Ltd.

Demo of Postman for REST

- Postman is a chrome app that is useful for REST testing

- Along with paid upgrades for JetPack - it can run tests in a group and offer Javascript response validation

Aprameyah
Technologies Pvt. Ltd

Data Binding Type Conversions

- Primitives automatically handled
- Params can be java classes that have a constructor with a string param
- Serializable java beans
- We can define custom converter classes for types that don't fit above definition

Aprameyah Technologies Pvt. Ltd.

Lets Try It

- Implement User Creation using automatic data binding

```
@RequestMapping(method=RequestMethod.POST, value="/",
        consumes=MediaType.APPLICATION_JSON_VALUE,
        produces=MediaType.APPLICATION_JSON_VALUE)
public Map<String,String> addUsers(@RequestBody User u) throws Exception{
```

- Implement User Update

```
@RequestMapping(method=RequestMethod.PUT, value="/",
        consumes=MediaType.APPLICATION_JSON_VALUE,
        produces=MediaType.APPLICATION_JSON_VALUE)
    public Map<String,String> updateUsers(@RequestBody User u) throws Exception{
```

- Implement Delete

```
    @RequestMapping(method=RequestMethod.DELETE, value="/{uid}",
            produces=MediaType.APPLICATION_JSON_VALUE)
    public Map<String,String> deleteUsers(@PathVariable("uid") Integer id) throws
Exception{
```

- Implement User retrieval with path params /users/307

Aprameyah Technologies Pvt. Ltd.

JAX-RS Comparison
Resources & Sub-Resources

```
@Path("/maps")
public class MapsResource {


    @Path("directions")
    public DirectionsResource getDirectionsResource() {
        return new DirectionsResource();
    }

    @GET
    @Produces("application/xml")
    public Map get() { ... }
    }
}

public class DirectionsResource {

    @GET
    public Response get() { ... }

    @PUT
    @Path("{version}")
    public void put(...) {
        ...
    }
}

/services/maps/directions/
```

Content Negotiation

- Within large organizations or on the Internet, SOA applications need to be flexible enough to handle and integrate with a variety of clients and platforms.

- Different clients need different formats in order to run efficiently. Java clients might like their data within an XML format. Ajax clients work a lot better with JSON.

- Clients will need content in different languages

Aprameyah Technologies Pvt. Ltd.

Http Content Neg

```
GET http://example.com/stuff
Accept: application/xml, application/json
```

- the client is asking the server for /stuff formatted in either XML or JSON. If the server is unable to provide the desired format, it will respond with a status code of 406, “Not Acceptable.”

- Accept: text/*;q=0.9, */*;q=0.1, audio/mpeg, application/xml;q=0.5

```
Preference order:

audio/mpeg
text/*
application/xml
*/*
```

- Accept: text/*, text/html;level=1, */*, application/xml

```
Preference order:

text/html;level=1
application/xml
text/*
*/*
```

Aprameyah
Technologies Pvt. Ltd.

Language Neg

- Accept-Language: en-us, es, fr
    - Requesting content in english, spanish and french
- Accept-Language: fr;q=1.0, es;q=1.0, en=0.1
    - French or Spanish preferred, accepts english

Aprameyah
Technologies Pvt. Ltd.

Design Principle - Complexity In Data

- An important principle of REST is that the complexities of your resources are encapsulated within the data formats

- Location information (URIs) and protocol methods remain fixed, data formats can evolve.

- Since complexity is confined to your data formats, clients can use media types to ask for different format versions.

Aprameyah
Technologies Pvt. Ltd.

Versioning

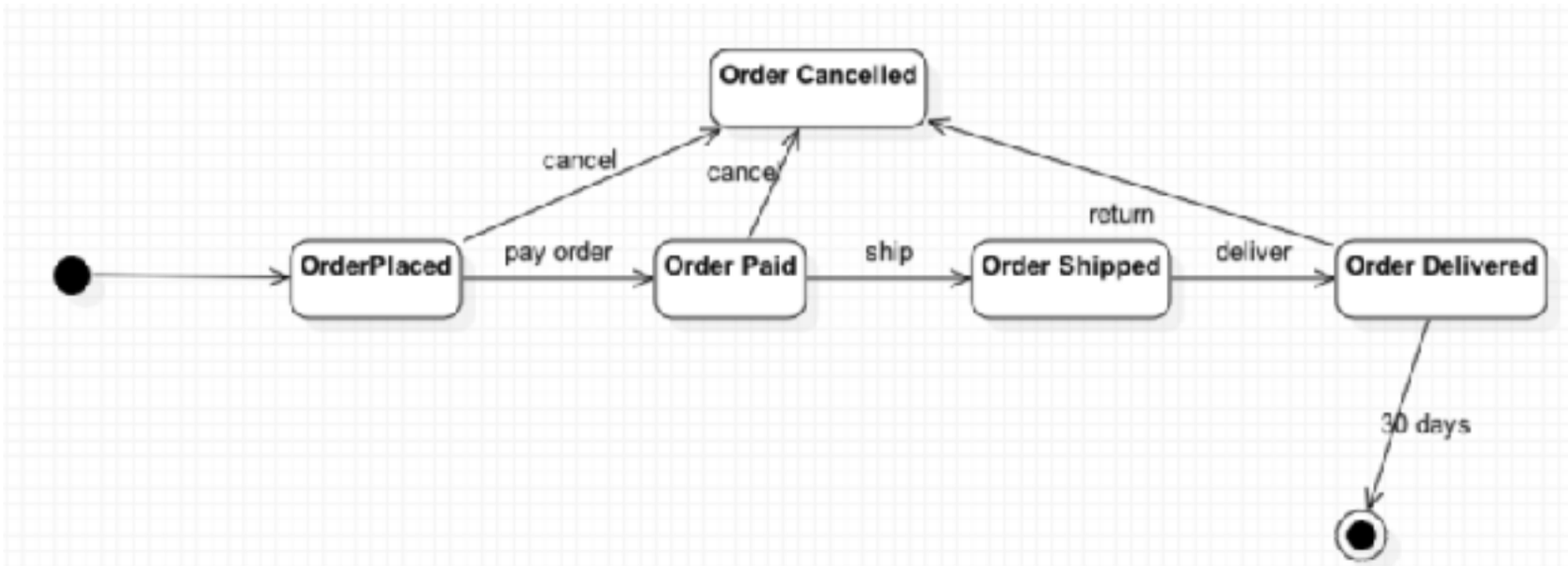
- One way to version REST is using different urls for different versions
    - http://host/services/v1/users and http://host/services/v2/users
- Another way to do it using custom content types on the same url
    - application/vnd.myapp.users-v1+json and application/vnd.myapp.users-v2+json
- Versionless extensibility driven with HATEOAS

Aprameyah
Technologies Pvt. Ltd.

HATEOAS

- location transparency. In a RESTful system that leverages HATEOAS, very few URIs are published to the outside world.
    - Clients only know logical names of resources - just like in a naming service
- Things like pagination work gracefully without the client knowing details of pagination params

Defining State Machine With HATEOAS



- States of a resource sometimes determine next available states

```
{
    order_id: 3,
    order_date: '2014-01-01',
    city: 'Bangalore',
    state: 'placed'
    links: [ {
         "rel": "cancel",
         "href": "http://localhost:8080/order/3?cancelled=true"
     },{
         "rel": "pay",
         "href": "http://localhost:8080/order/3?pay=true"
     } ]
}
```

Aprameyah
Technologies Pvt. Ltd.

Security

- Based on JEE Role based and principal based security
- Basic authentication flow shown below

```
GET /customers/333 HTTP/1.1

HTTP/1.1 401 Unauthorized
WWW-Authenticate: Basic realm="CustomerDB Realm"

GET /customers/333 HTTP/1.1
Authorization: Basic YmJ1cmtlOmdlaGVpbQ==
```

Aprameyah Technologies Pvt. Ltd.

Auth Configuration

```
<?xml version="1.0"?>
<web-app>
   <security-constraint>
      <web-resource-collection>
         <web-resource-name>users</web-resource-name>
         <url-pattern>/services/users</url-pattern>
         <http-method>POST</http-method>
      </web-resource-collection>
      <auth-constraint>
         <role-name>admin</role-name>
      </auth-constraint>
   </security-constraint>
   <login-config>
       <auth-method>BASIC</auth-method>
       <realm-name>jaxrs</realm-name>
   </login-config>
   <security-role>
       <role-name>admin</role-name>
   </security-role>
</web-app>
```

```
@Path("/users")
@RolesAllowed({"ADMIN"})
public class UserResource {
```

Aprameyah
Technologies Pvt. Ltd.

Programmatic Security

- When role based is not enough:
  - I can see my salary record not everyone else's

```
public interface SecurityContext {
    public Principal getUserPrincipal();
    public boolean isUserInRole(String role);
    public boolean isSecure();
    public String getAuthenticationScheme();
}
```

- Security context can be injected into the method or class

```
@Context SecurityContext sec
```