apigee

API First Development Workshop

Diego Zuluaga

Diego Zuluaga
dzuluaga@apigee.com
@dzuluaga

@apigee | #Apigee

Before we start …..

Install Node.js (LTS)

nodejs.org/en/download/

We work for Apigee

Why do we do this workshop?

Apigee Trial / Startup / Apigee SMB

	edge Trial	edge Startup	edge SMB
INCLUDED SERVICES			
Deployment	Apigee Cloud	Apigee Cloud	Apigee Cloud
API Calls	Up to 1 million	5 million per quarter	25 million per quarter
Cost	Free	from $300/month	from $2250/month
Support	Community support only	1 support account	1 support account

apigee

Who are you?

Sign up for Apigee

http://accounts.apigee.com/accounts/sign_up

And don't forget to join

http://community.apigee.com

Home of more than 4K API practitioners

Get answers, ideas and support from the Apigee Community

http://community.apigee.com

Agenda

Section 1

RESTful API Best Practices

Section 2

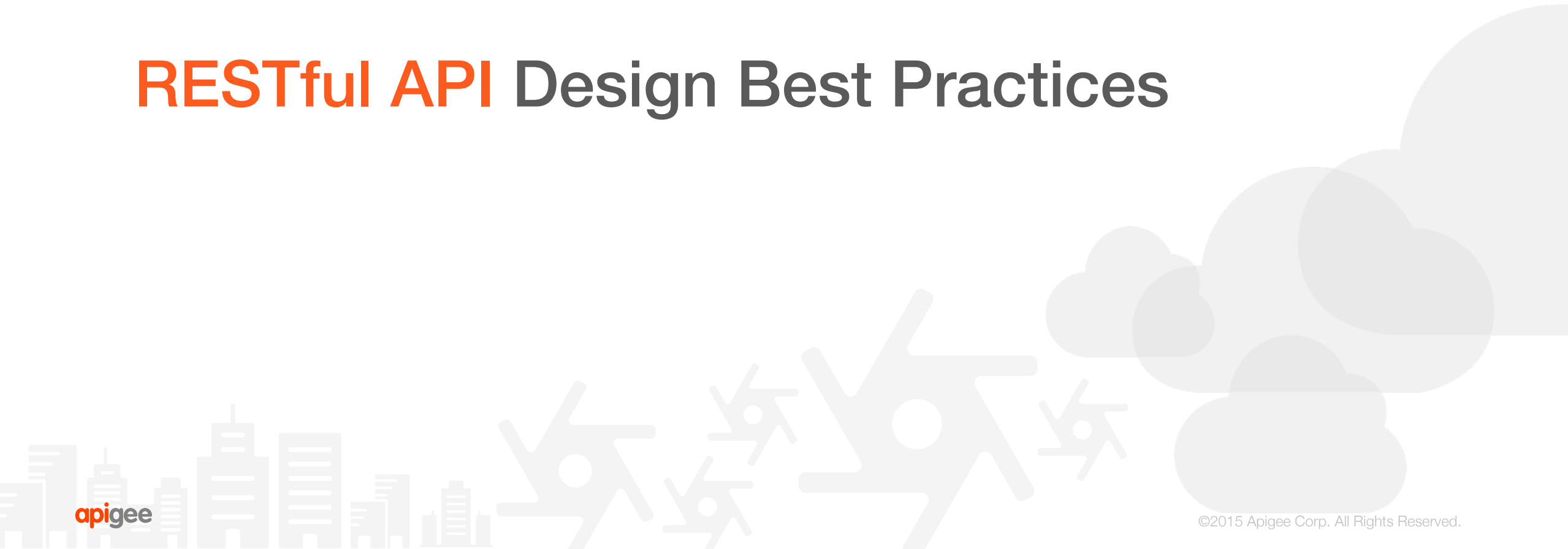
API-First Development

Section 3

Let’s get our hands dirty -- workshop

apigee

RESTful API Design Best Practices



In the beginning…

Document: <draft-box-http-soap-00.txt> September 1999
Category: Informational

SOAP: Simple Object Access Protocol

XML-RPC Specification

Tue, Jun 15, 1999; by Dave Winer.

Updated 6/30/03 DW

Updated 10/16/99 DW

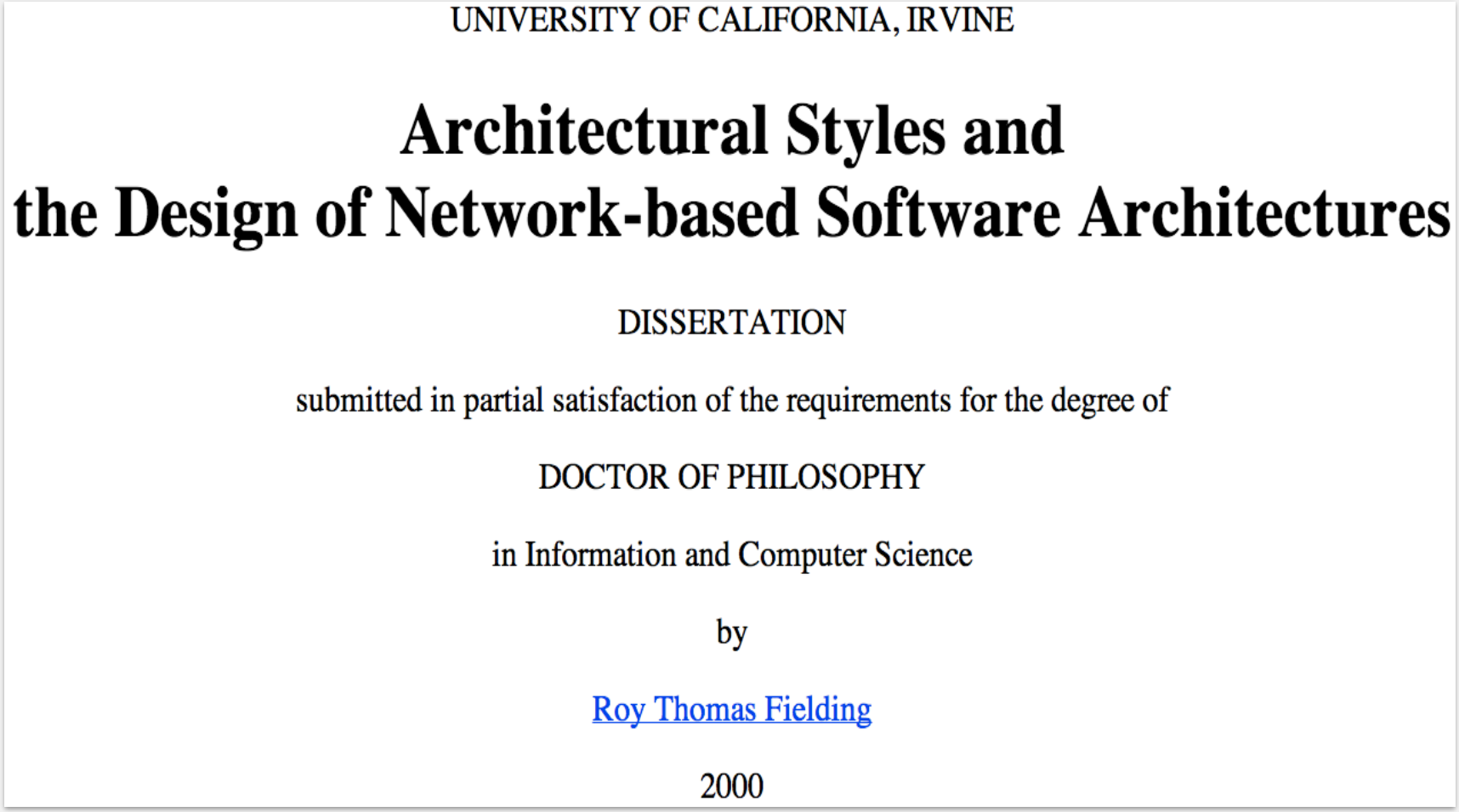
Updated 1/21/99 DW

This specification documents the XML-RPC protocol implemented in UserLand Frontier 5.1.

SOAP and XML-RPC

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope
        xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <soapenv:Header>
    <ns1:RequestHeader
         soapenv:actor="http://schemas.xmlsoap.org/soap/actor/next"
         soapenv:mustUnderstand="0"
         xmlns:ns1="https://www.google.com/apis/ads/publisher/v201403">
      <ns1:networkCode>123456</ns1:networkCode>
      <ns1:applicationName>DfpApi-Java-2.1.0-dfp_test</ns1:applicationName>
    </ns1:RequestHeader>
  </soapenv:Header>
  <soapenv:Body>
    <getAdUnitsByStatement xmlns="https://www.google.com/apis/ads/publisher/v201403">
      <filterStatement>
        <query>WHERE parentId IS NULL LIMIT 500</query>
      </filterStatement>
    </getAdUnitsByStatement>
  </soapenv:Body>
</soapenv:Envelope>
```

The dawn of REST



UNIVERSITY OF CALIFORNIA, IRVINE

Architectural Styles and the Design of Network-based Software Architectures

DISSERTATION

submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in Information and Computer Science

by

Roy Thomas Fielding

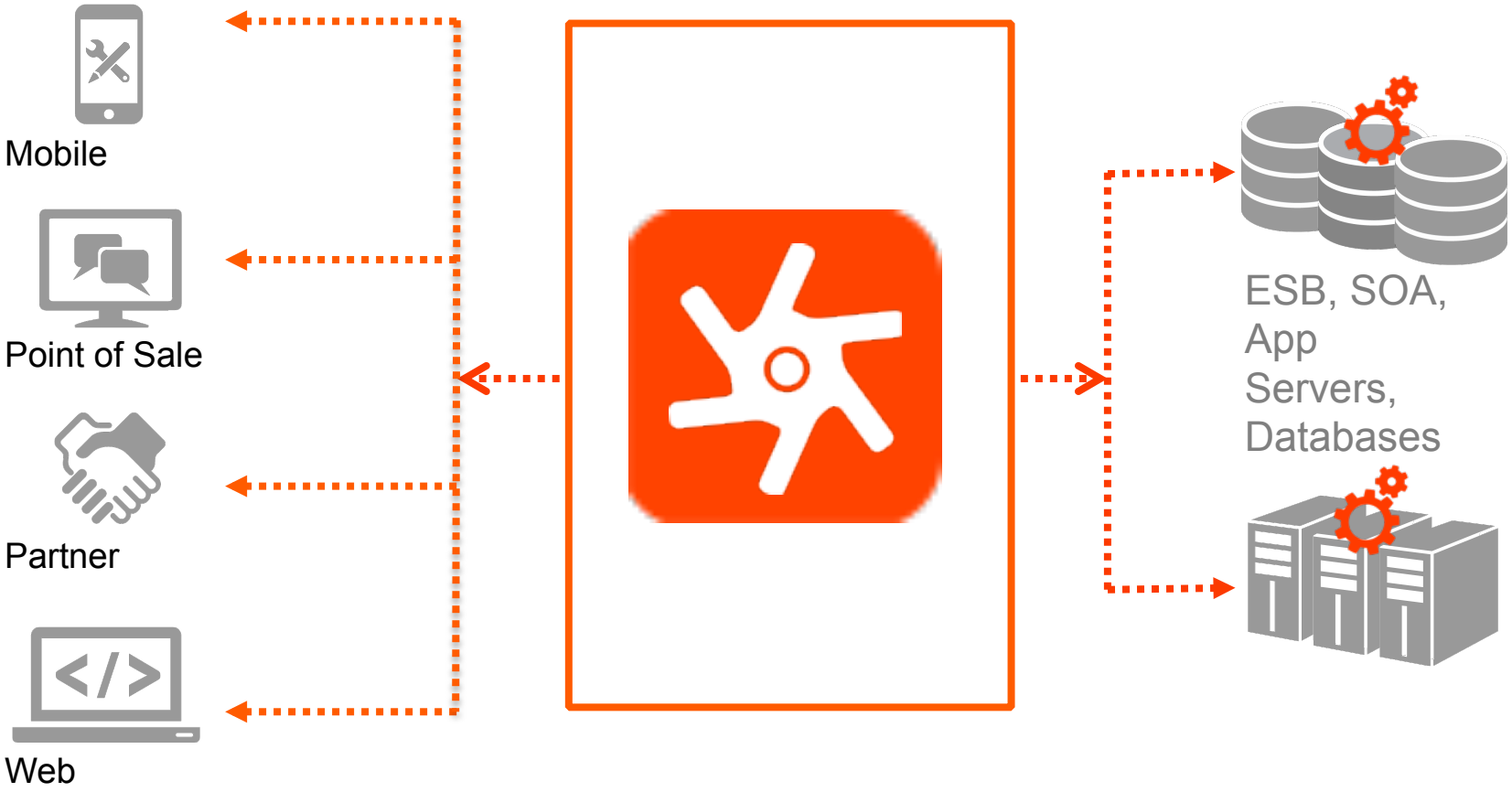
2000

REpresentational State Transfer

- Resources and resource identifiers
- Transport and semantic independence
- Statelessness
- Interface/contract uniformity
- Metadata and shared understanding of data types
- Self-descriptive messages
- Hypertext as the Engine of Application State (HATEOAS)

What could be easier?

GET https://www.googleapis.com/adexchangebuyer/v2/accounts



Teaching a dog to REST

Reluctant API Design

```
/getDog
/getAllDogs
/petDog
/feedDog
/createRecurringDogWalk
/giveCommand
/healthCheck
/getRecurringDogWalkSchedule
/getLocation
/teachTrick
```

Reluctant API Design

/getAllDogs

/petDog

/feedDog

/createRecurringWakeUp

/giveCommand

/checkHealth

/getRecurringWakeUpSchedule

/getLocation

/getDog

/newDog

/getNewDogsSince

/getSittingDogs

/setDogStateTo/saveDog

/getAll**Leashed**Dogs

/verify**Veterinarian**Location

/createRecurring**Medication**

/doDirect**Owner**Discipline

/doCheckupWith**Veterinarian**

/getRecurring**Feeding**Schedule

/get**Hunger**Level

/get**Squirrels**ChasingPuppies

/newDogFor**Owner**

/getNewDogsAt**Kennel**Since

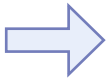
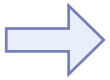
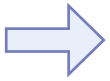
/getSittingDogsAt**Park**

/set**Leashed**DogStateTo

/save**MommaDogs**Puppies

apigee

It happens in the real world

API	NVP	SOAP
AddressVerify	NVP	SOAP
BillOutstandingAmount	NVP	SOAP
Callback	NVP	
CreateRecurringPaymentsProfile	NVP	SOAP
DoAuthorization	NVP	SOAP
DoCapture	NVP	SOAP
DoDirectPayment	NVP	SOAP
DoExpressCheckoutPayment	NVP	SOAP
DoNonReferencedCredit	NVP	SOAP
DoReauthorization	NVP	SOAP
DoReferenceTransaction	NVP	SOAP
DoVoid	NVP	SOAP
GetBalance	NVP	SOAP
GetBillingAgreementCustomerDetails	NVP	SOAP
GetExpressCheckoutDetails	NVP	SOAP
GetRecurringPaymentsProfileDetails	NVP	SOAP

apigee

Design for adoption

Resource modeling

We only need two base URLs for a resource

A collection `/dogs`

A resource `/dogs/charlie`

Three’s company

```
/owners/diego/dogs
```

Sweep complexity behind the “?”

```
/dogs?color=red&state=running&location=park
```

Actions are resources

(hypothetical)

```
/convert?from=EUR&to=CNY&amount=100
```

DigitalOcean

```
/droplets/{droplet_id}/reboot
```

Facebook

```
/search?q=watermelon&type=post
```

Never EVER

Never Break
The Client.

EVER.

unless…

Versioning schemes

```
/2010-04-01/Accounts/
```

```
/services/data/v30.0/limits
```

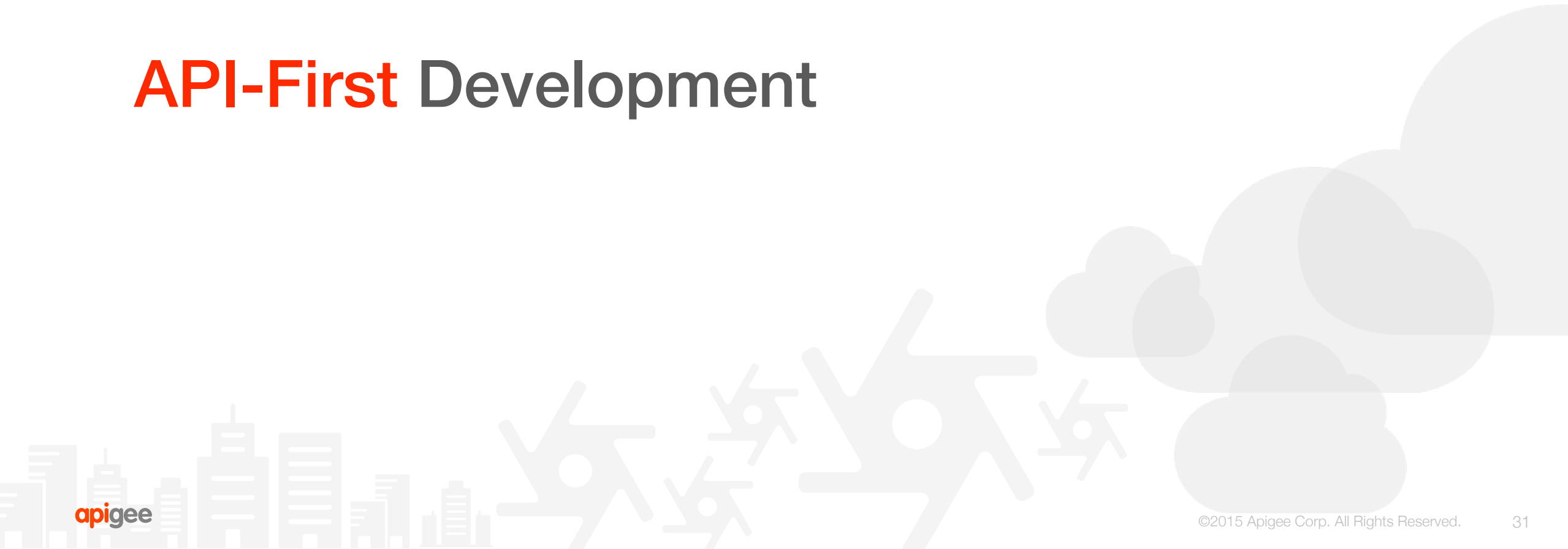
```
/v2.0/me
```

```
/v2/users/{userid}/checkins
```

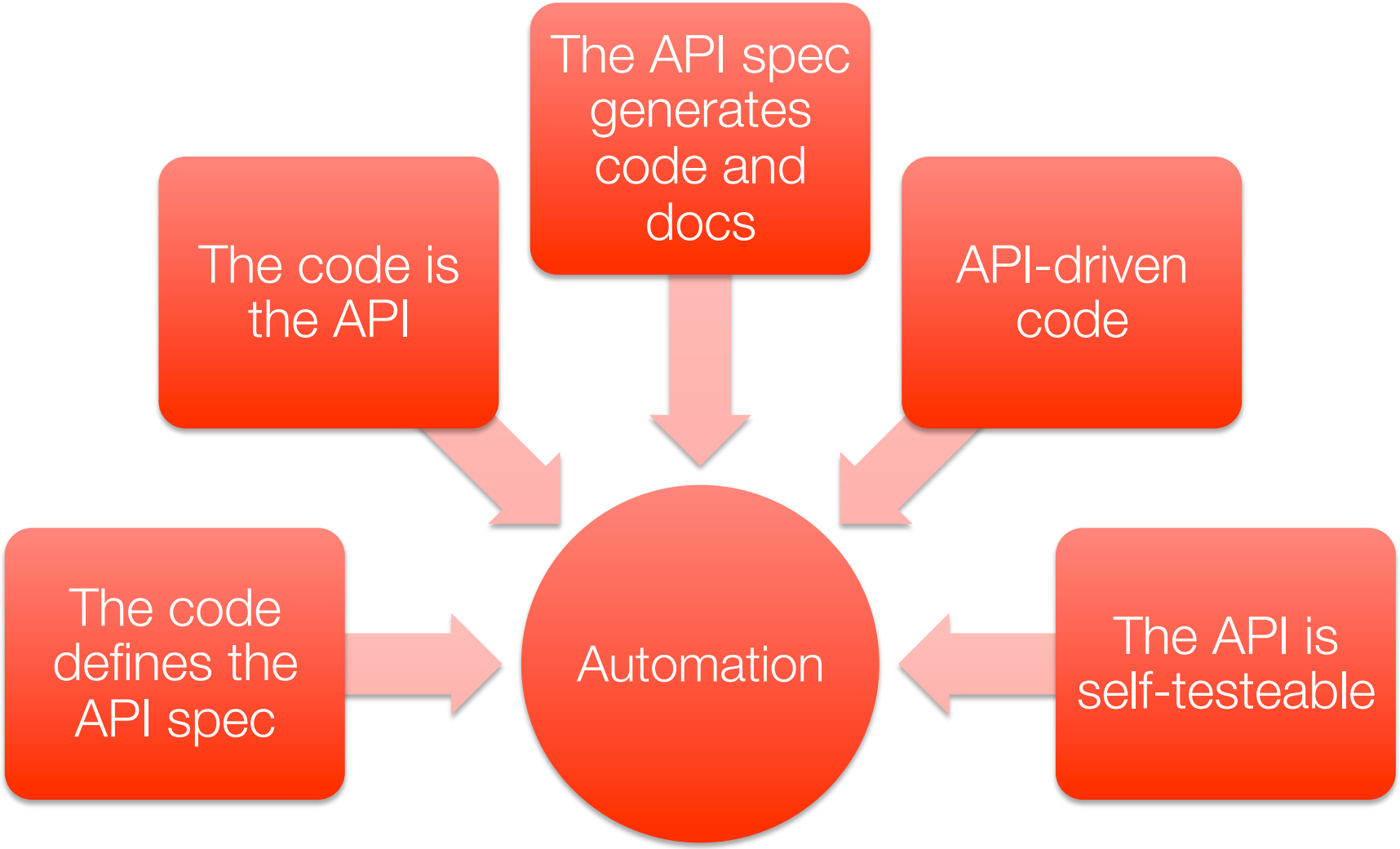
apigee

apigee

API-First Development



Zen of API Development



The code defines the API Spec

API Spec is either manually or automatically generated

Annotation or data driven

Maintained in code

Extensible by vendor extensions

Take baby steps… then automate

```
@Path ("/my-resource")
@Api (value="/my-resource",
    description="Rest api for do operations on admin",
    produces=MediaType.APPLICATION_JSON)
@Produces ({ MediaType.APPLICATION_JSON })
class MyResource {
    @ApiOperation(value = "Get specific element",
        httpMethod = "GET",
        notes = "Fetch the selement of the collection",
        response = Response.class)
    @ApiResponses(value = {
    @ApiResponse(code = 200, message = "Element found"),
    @ApiResponse(code = 404, message = "Element not found"),
    @ApiResponse(code = 500, message = "Server error due to encoding"),
    @ApiResponse(code = 400, message = "Bad request: decoding error"),
    @ApiResponse(code = 412, message = "Prereq: Required data not found")
    })
    public Response get(

    @ApiParam(value = "UUID of the element", required = true)
    @PathParam("uuid") Sting uuid)
```

apigee

The code is the API

Interpreted

No formal specification

Map to business logic

```
// GET method route
api.get('/products', function(req, res, next) {
  res.send('GET request to products collection'); // retrieve
})

// POST method route
api.post('/products', function(req, res, next) {
  res.send('POST request to products collection'); // create resource
})
```

Based on plugins or middlewares (logger, body parser, compression, error handler, response-time, etc.)

The API generates code, docs, and tests

Interface Definition Language (IDL) defines the API

Client and server SDKs and side stubs code stubs are generated, tested and semantic-versioned

Examples: SOAP, CORBA, and similar RPC systems

HTML docs (SmartDocs), Postman Collections, cURL commands, etc.

And tests…

The API Spec is the documentation

Everybody embraces ;-) multi-vendor support

Publish API Docs and execute them anywhere.

https://github.com/OAI/OpenAPI-Specification/wiki/Sites-and-Services

Sites and Services

Jeshan Giovanni BABOOA edited this page 21 days ago · 26 revisions

Service Providers and Infrastructure Providers

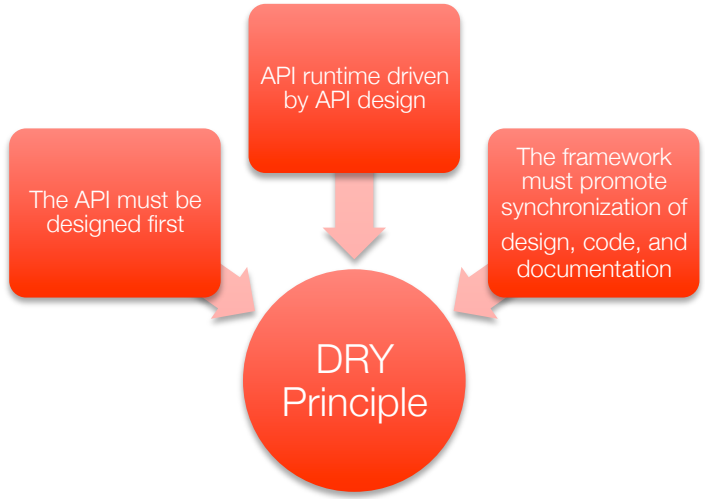
The following provide Swagger support to their software or services.

- 3Scale Active Docs
- SwaggerGo!
- Amazon API Gateway
- API2Cart
- Apigee Smartdocs
- APIMATIC Codegen
- Apache Apollo
- Apache Camel
- Api Spark
- Appcelerator
- Axway
- Banckle - docs
- Bench Accounting
- Bitdango
- BitMEX
- Catch Software
- Callfire
- Concur
- Data2CRM.API
- DataFire
- Runscope API Monitoring and Testing
- Sandbox
- Sensr.net
- SnapLogic SwaggerSnap
- SoapUI - Support via a plugin which is not compatible with SoapUI 5.1 and later. See Deprecated SoapUI Extension Plugins.
- Strongloop
- SubtleData
- Subledger
- Tibco API Exchange
- Wordnik
- Nomos Software Jax-RS Server Generation
- Meruvian our work also in github.com/meruvian/yama take a look!
- Tyk Cloud

Sites and Applications

- Hivepod
- Eyefi
- Marvel Comics Developer Portal
- Kixeye
- US Government
- NPR
- Klout
- IGN
- snapCX

API-driven code philosophy



The API must be designed first.

The artifact that represents the API design must drive the API runtime.

The API design will change, and the framework must make it possible to adapt quickly without letting the code, design, and documentation fall out of sync.

The "DRY" Principle

API-First Philosophy

The API must be designed first.

API-First philosophy

The API must directly drive runtime and documentation.

API-First Philosophy

API design, documentation, and code must remain in sync.

API-First Philosophy

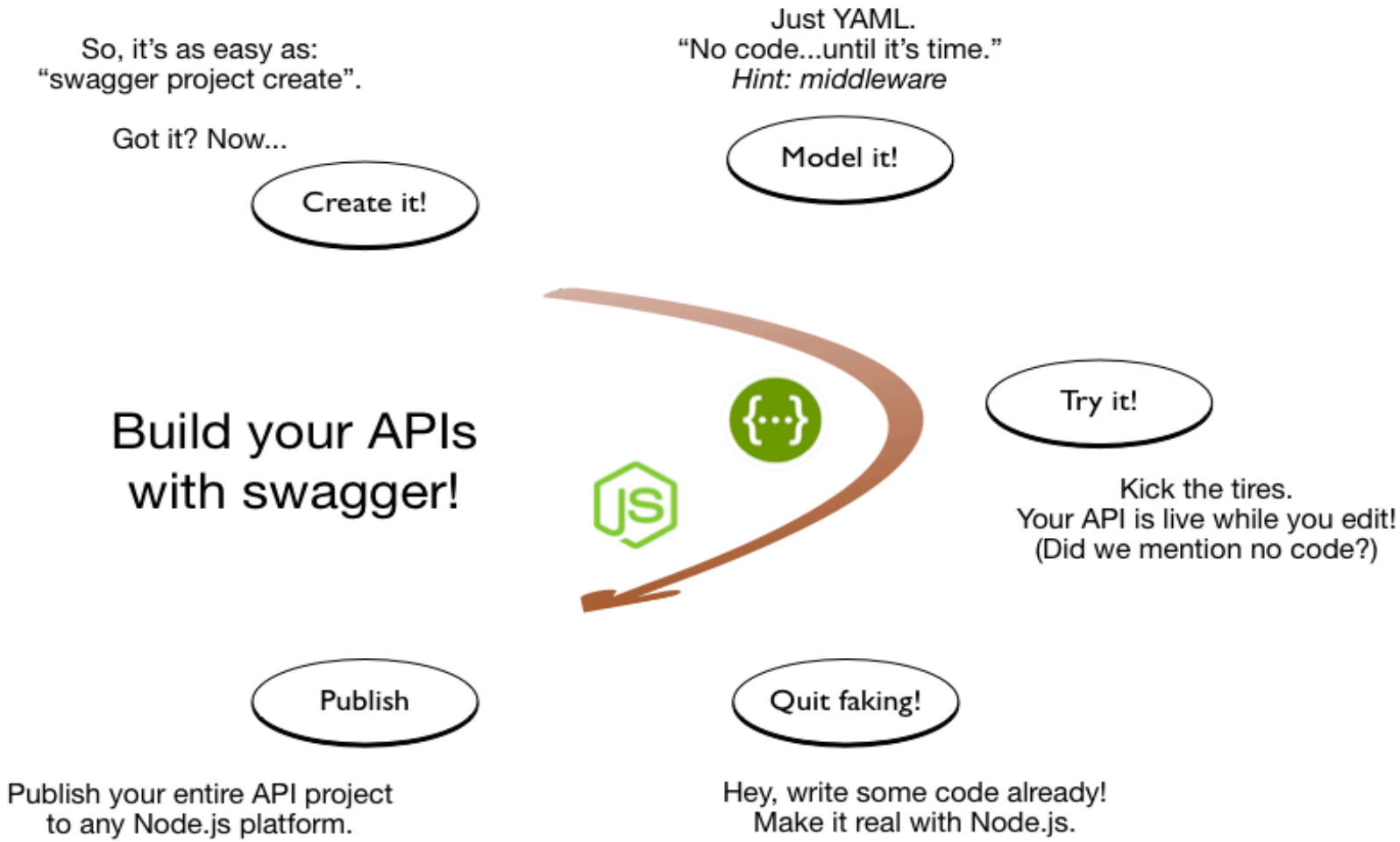
Overall…

The system must adhere to the "DRY Principle”

apigee

Swagger-Node

Swagger-Node: Development Lifecycle



Swagger-Node

The API is written in Open API aka Swagger using API Studio

The Swagger API document is parsed when server starts

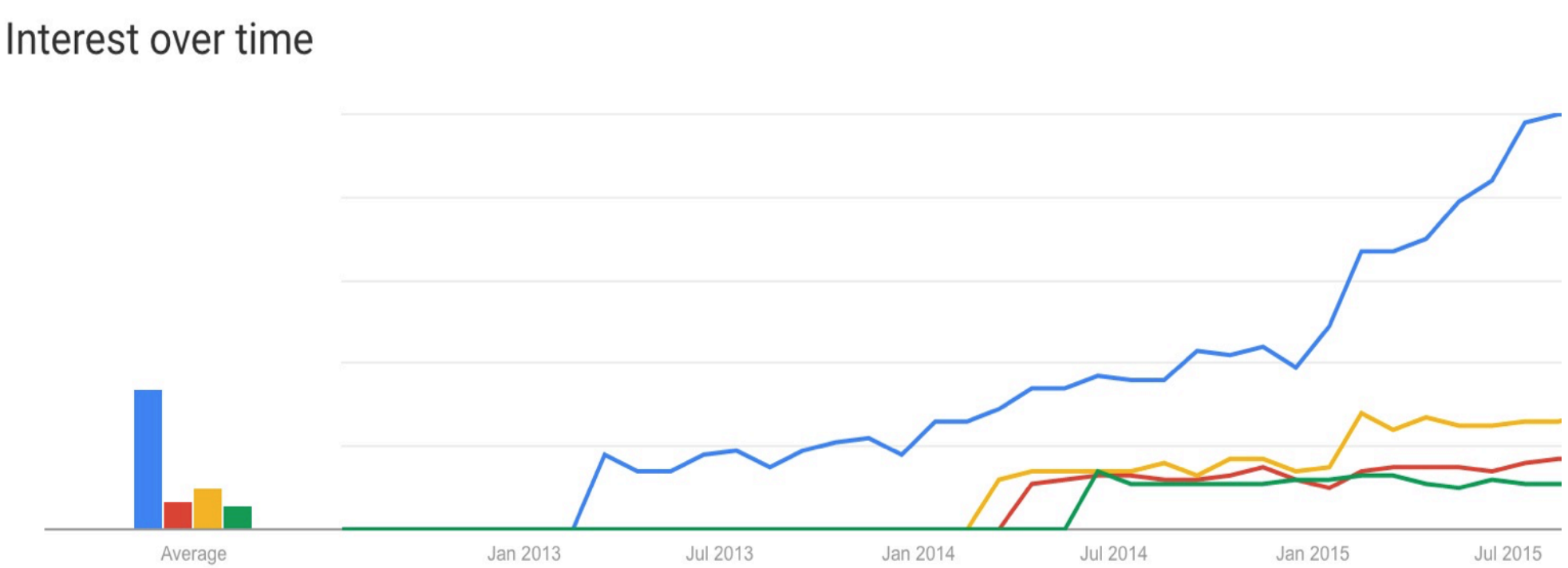
Incoming calls are classified, validated, and routed in real time

Integrates with Connect, Express, Hapi, Restify, Sails...

Incorporates a plugin model for Swagger (or non-Swagger) extensions

But why Swagger/Open API ?

swagger api | raml api | api blueprint | wadl api



But why Node.js ?

Installation

Installation – Step 1

Make sure you have node.js installed. `V4.2.x` preferred.

https://nodejs.org/en/

```
$ node -v

//The latest LTS version 4.2.x
```

Installation – Step 2

Install

```
$ sudo npm install -g swagger
```

Verify:

```
$ swagger --version

    0.7.4 or later
```

What do you get?

- CLI
  - project scaffolding
  - project lifecycle
- Your own API Studio (sort of :-))
  - Write YAML
  - Immediate feedback loop
  - Try-it on the fly
- Runtime

Worskshop Source

https://github.com/apigee/design-first-api-dev-workshop

Let's take a tour of API Studio

- Code Completion
- Immediate feedback loop
- Simulated Response aka "Mock Mode"
- Collaboration
- Download YAML/JSON & Node.js project
- Generated doc
- Raw Spec endpoint

Let's create a project

```
$ swagger project create {project_name}
```

Let's run the project

```
$ cd $project-name

$ swagger project start
```

Let’s make some API call

```
$ curl http://127.0.0.1:10010/hello?name=Scott
```

Let’s not forget the tests

```
$ swagger project generate-test

$ swagger project test
```

How about editing ?

```
$ swagger project edit
```

Dissecting it

- Project Conventions
- swagger spec
- controllers
- helpers

Let’s add something new

- A new path that uses POST operation

..which requires

- A new controller

Try it out

```
$ curl -X POST http://127.0.0.1:10010/conf/add -H
"content-type:application/json" -d '{"x":5,"y":6}'
```

Deploy to Apigee

```
$ sudo npm install -g apigeetool
```

```
$ apigeetool deploynodeapp -u sdoe@apigee.com -o sdoe
-e test -n 'Test Node App 2' -d . -m app.js -b /node2
```

OR

```
$ a127 project deploy
```

How about some API Management

- Let’s add quota

Quota : Get the bits

- Let's add quota *[This is subject to change. See latest at https://github.com/apigee-127/volos-swagger-apply/blob/master/README.md]*

```
npm install --save volos-swagger-apply
npm install --save volos-quota-memory
mkdir api/fittings
```

Quota : Annotate your Swagger

```
x-volos-resources:
 MyQuota:
   provider: volos-quota-memory
   options:
     timeUnit: minute
     interval: 1
     allow: 1

paths:
  /hello:
    x-volos-apply:
        MyQuota: {}
```

Quota : Tell the framework about it.

Add fitting to config/default.yaml to `swagger_controllers` :

```
swagger_controllers:
     - onError: json_error_handler
     - cors
     - swagger_security
     - _swagger_validate
     - express_compatibility
     - volos-swagger-apply                    # <- RUN HERE
     - _router
```

Quota : Verify that it works

```
curl -X POST http://127.0.0.1:10010/conf/add -H
"content-type:application/json" -d '{"x":5,"y":6}'

{"message":"exceeded quota","status":403}
```

Where can I get help?

Community: https://community.apigee.com

Github:

- github.com/apigee
- github.com/apigee-127
- github.com/swagger-api
- github.com/dzuluaga/datamodel-to-oas

apigee

Thank you