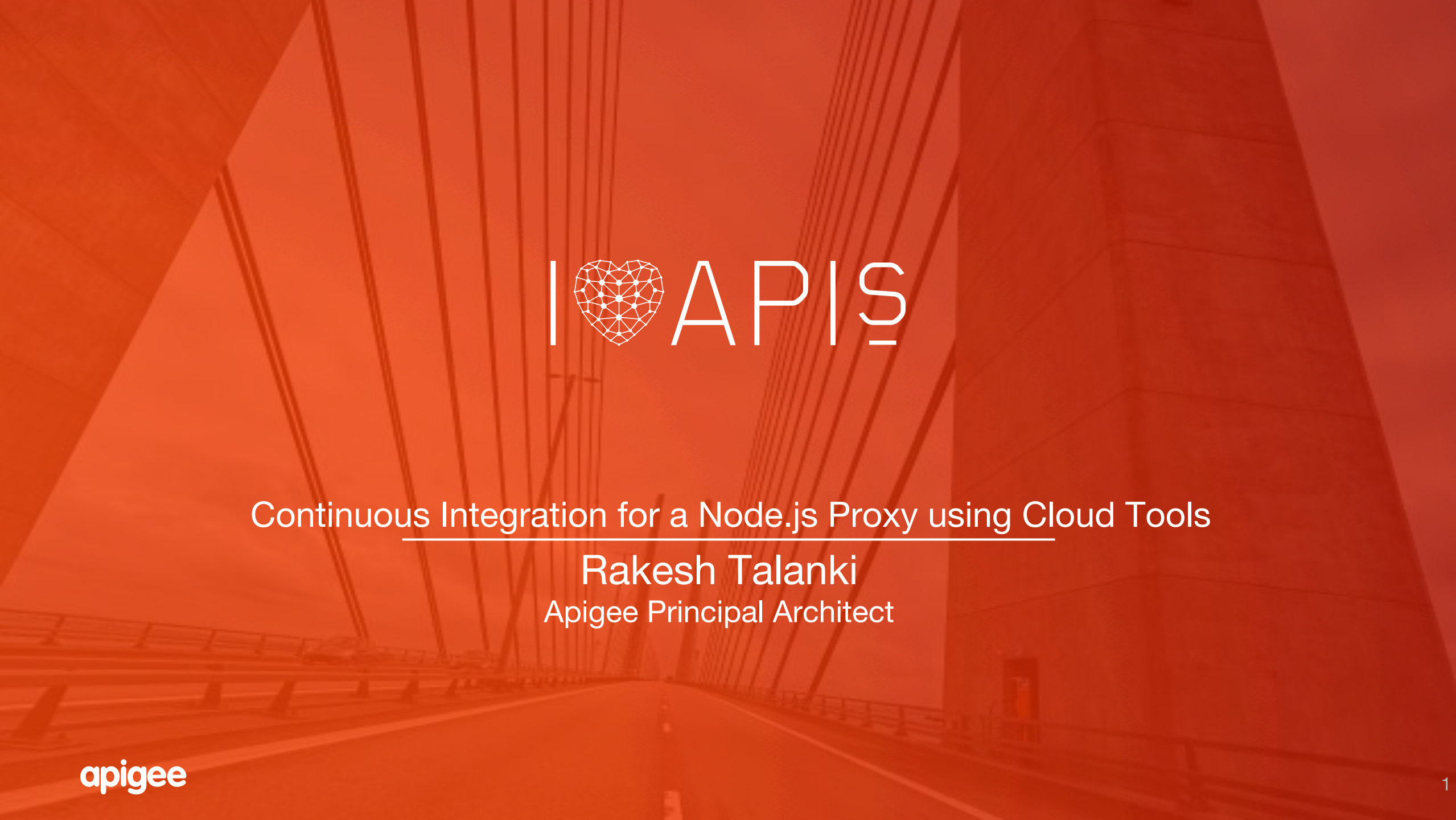


Continuous Integration for a Node.js Proxy using Cloud Tools

Rakesh Talanki
Apigee Principal Architect

apigee

Introduction and Agenda

Agenda

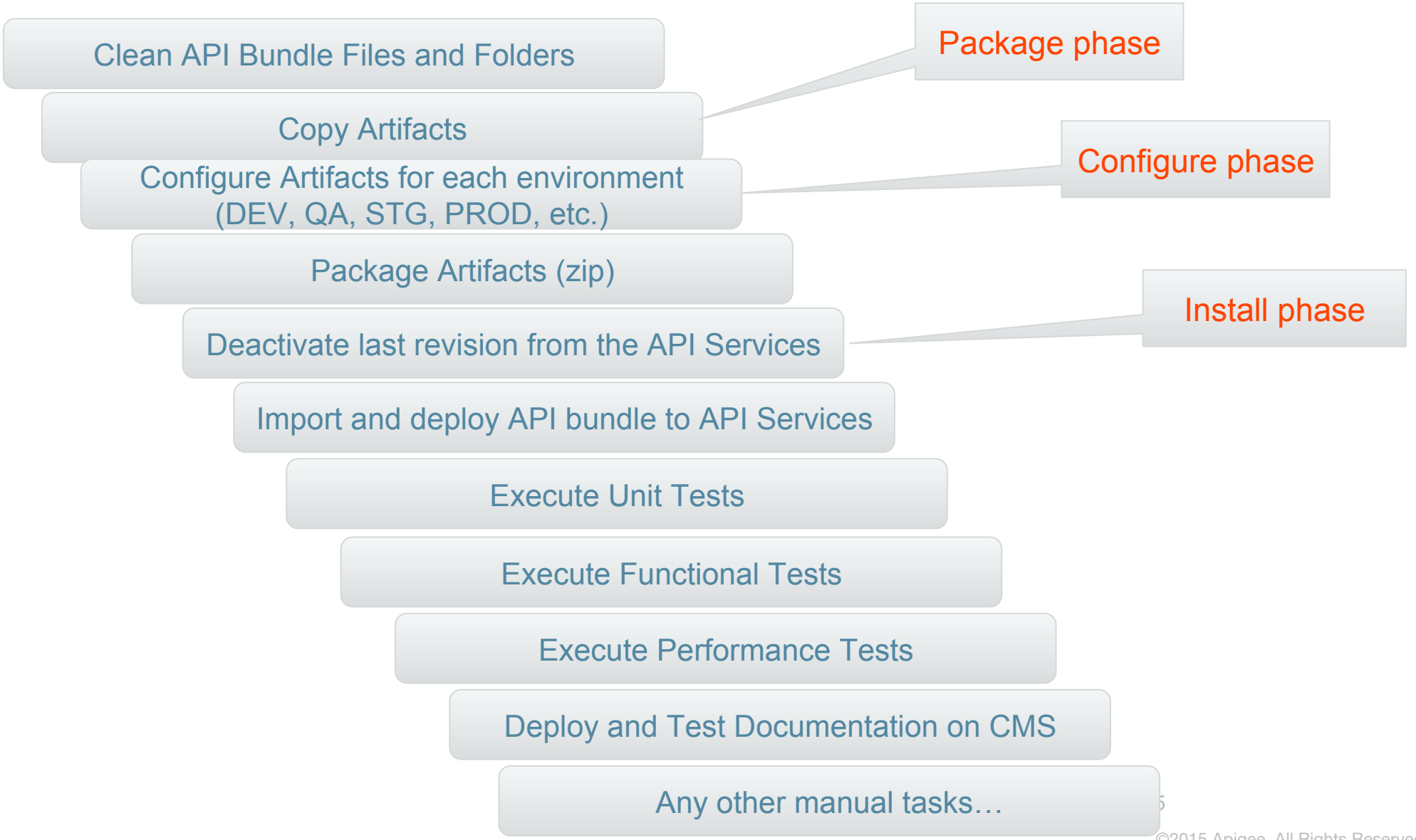
Set up Continuous Integration for API's using Cloud Tools

1.	Set up Node.js API in Apigee	8.00
2.	Build API scaffolding with Yeoman	15:00
3.	Use Grunt to build and deploy	10:00
4.	Use Git for Source Control	5.00
5.	Test using Postman	10:00
6.	Test using Mocha, Chai and Nock	10:00
7.	API Documentation	5:00
8.	Use Travis to set up CI	30:00

Getting ready

- Apigee Free Account
- NPM and Node.js - https://goo.gl/080g8Q
- Git (optional) - https://goo.gl/rTyIvP
- Github Free Account – http://www.github.com
- Apigee Free Account on Cloud
- Travis Account (free)- https://travis-ci.org/

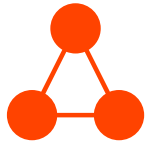
There must be a way to automate all of tedious work!



Before

After

Continuous Integration



Development

SCM

Build

SIT

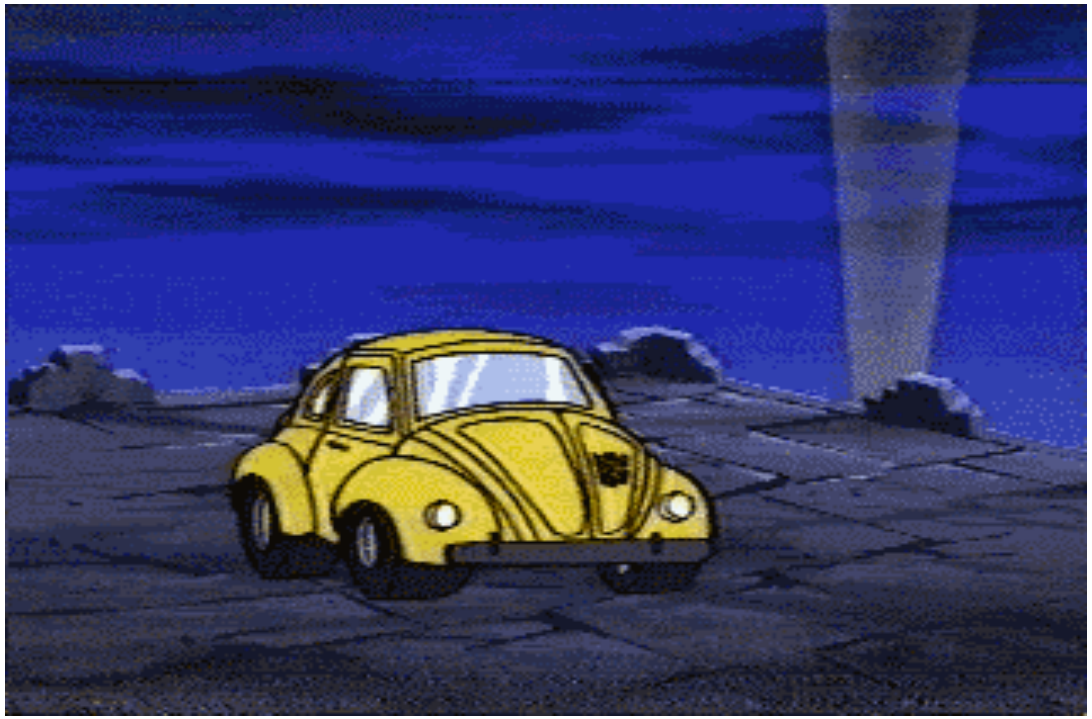
UAT

Prod

Continuous Integration (CI) is the practice, in software engineering, of merging all developer working copies with a shared mainline several times a day

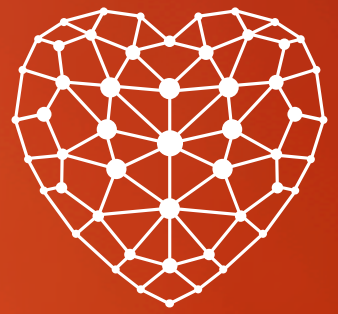
CI: The Volkswagen Approach

We master Emission Test Results with Software



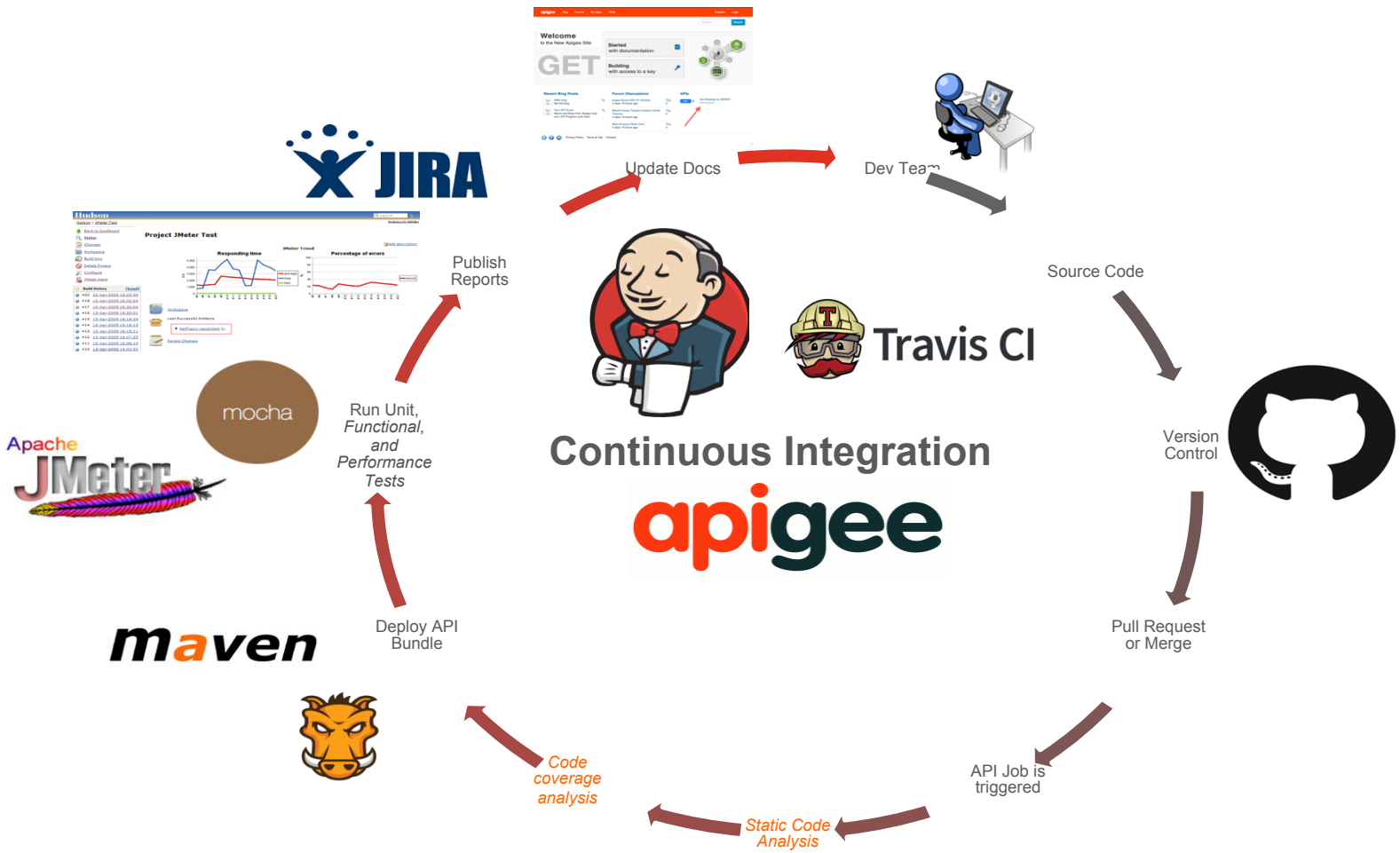
Then, we detect when our tests are being run in a CI server, and make them pass.*

*https://github.com/auchenberg/volkswagen



Building a CI Environment

CI: The Process

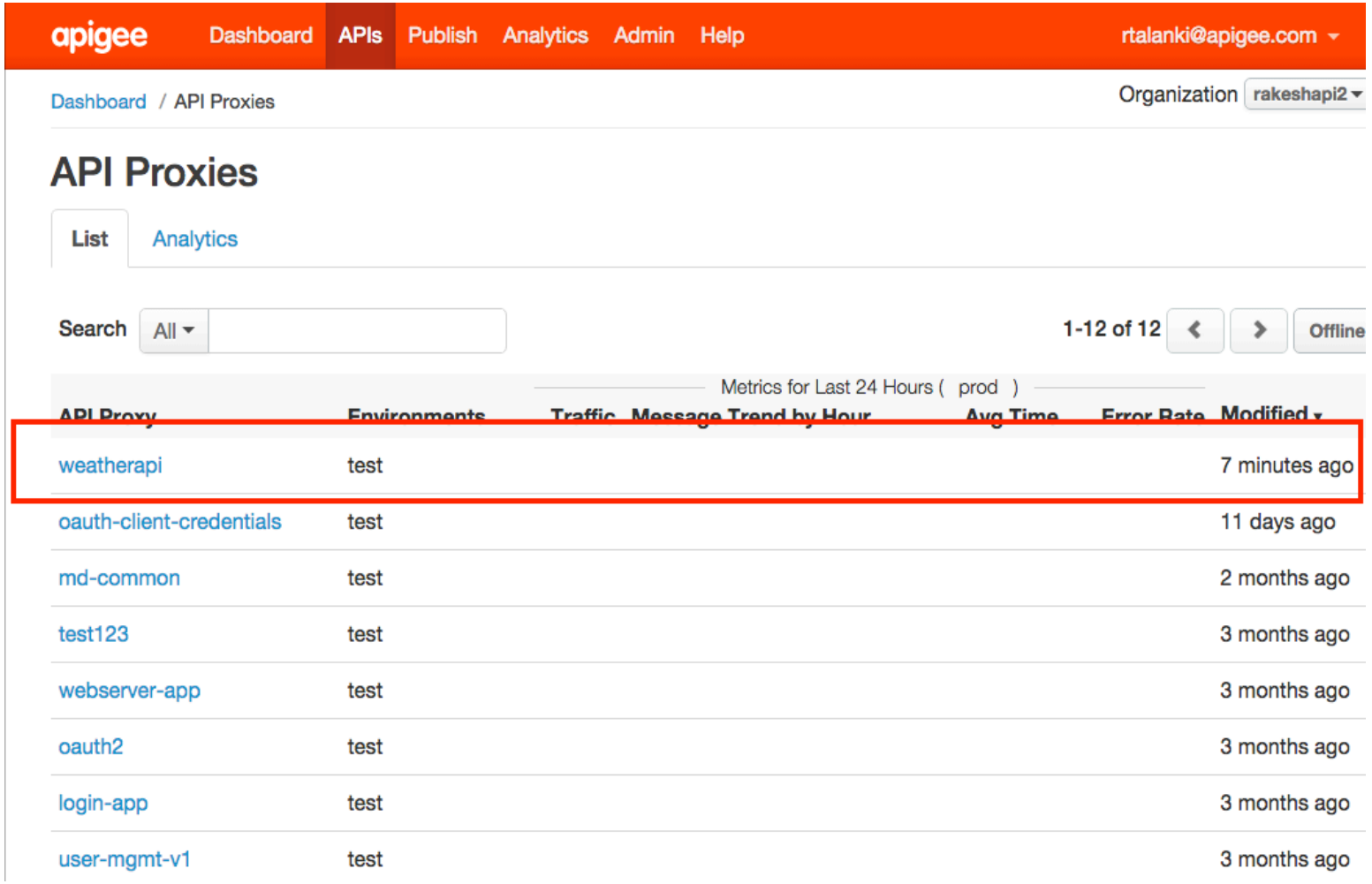


Step 1: Build an API Proxy

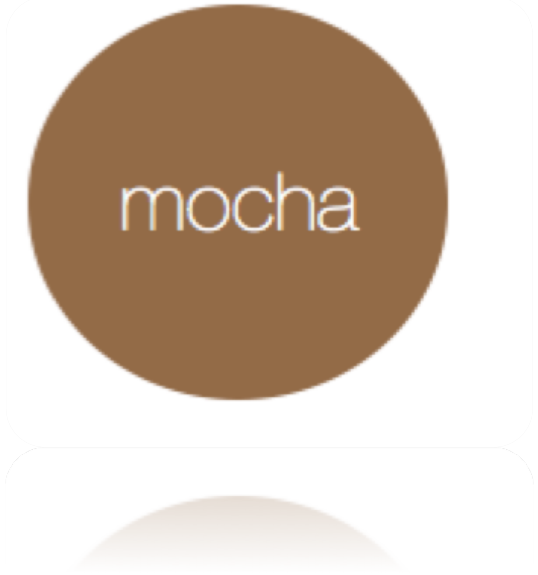
Step 2: Decide on a Build Tool

maven

Step 3: Deploy to Apigee on Public Cloud

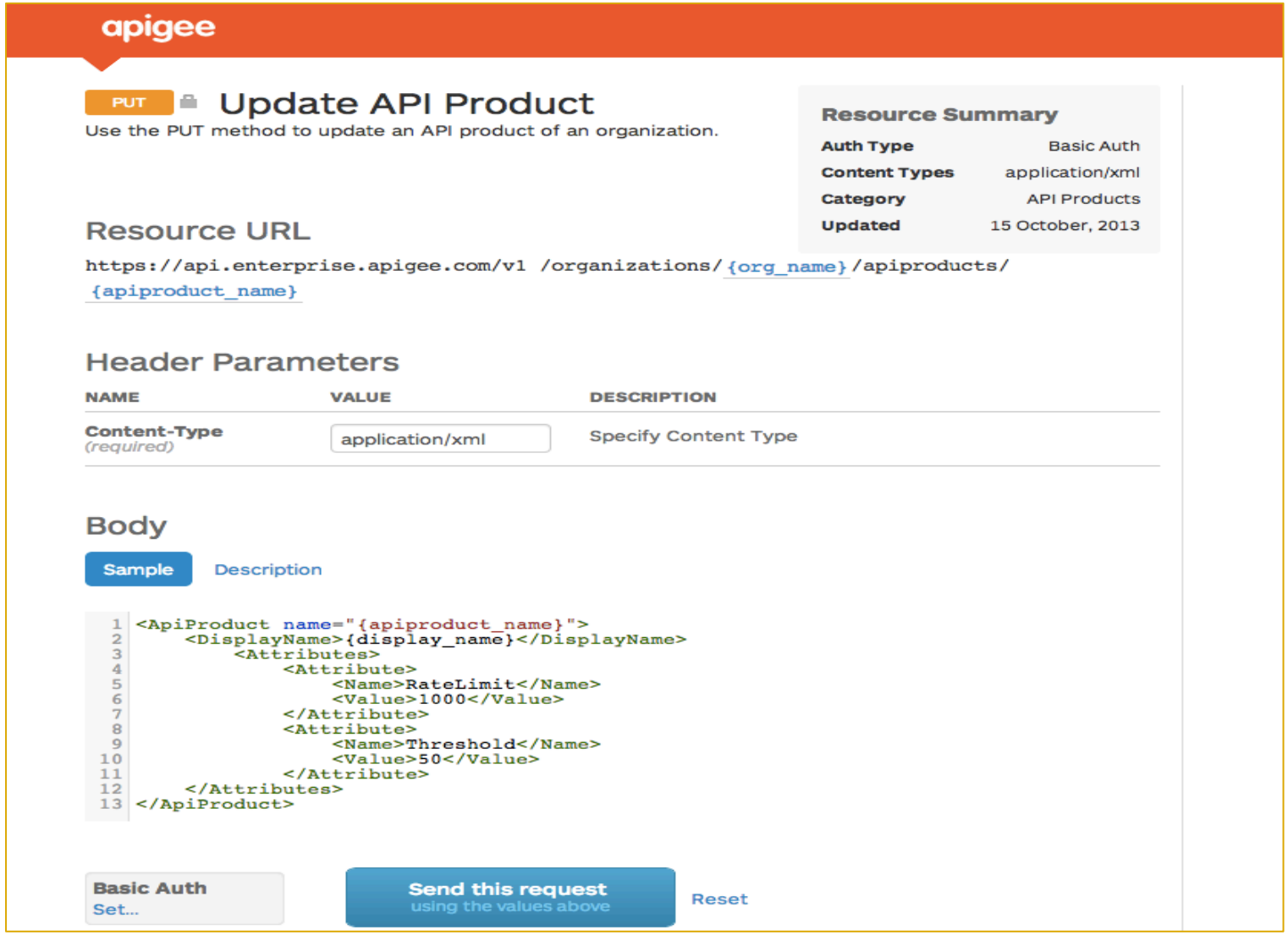


Step 4: Run Local Tests



Step 5: Use a CI Tool

Step 6: Generate Interactive Documentation



Hands-on

Putting it all together: Continuous Integration

API Proxy Scaffolding Generator

- Gets you started with starter API Proxy
- Standardizes naming conventions by generating policies and other artifacts
- Based on Yeoman, so it can extended for other tools, not only Grunt, but also Maven

API Proxy Scaffolding Generator

Let's try it!

Install Npm and Node: Open http://nodejs.org in a browser and click Install.

Check: node --version

Upgrade Node to 0.10.35 or higher:

npm cache clean

npm update –g

Install yeoman: npm install –g yo

Install Grunt: npm install –g grunt-cli

Install Grunt plugin: npm install -g generator-apigee-deploy-grunt-api

Run yo: yo apigee-deploy-grunt-api

http://goo.gl/lSZrth

Build and Deploy

Grunt or Maven?

Apigee Deploy Grunt Plugin

- TDD Ready - Mocha, Jasmine, Karma, any JS test framework and even Jmeter
- Supports code review - with JSHint and ESLint – cyclomatic complexity
- Supports Configuration Management - Search and replace based on XPath and RegExp
- Supports node.js remote deployment and Java Policies
- Plays well with CI (Continuous Integration) – Jenkins, Travis, Go, Bamboo, etc.
- Supports reusable policies via search and replace files and Git Submodules
- It's way easier to customize via Grunt Custom Tasks

Apigee Deploy Maven Plugin

- Test based on JMeter
- Config Management - Search and replace based on Xpath
- Plays well with CI – Jenkins mostly. Nice looking reports
- Supports node.js remote deployment and Java Policies
- Supports reusable policies via Supports Proxy Dependency Maven Plugin

Grunt API Lifecycle Management Plugin

- Easy and flexible
- It's Node! NPM
- Compatible with CI
- https://github.com/apigeecs/apigee-deploy-grunt-plugin
  - Follow steps from README.md
- Plays well with TDD frameworks
- Empowers the developer to apply continuous improvement to the lifecycle

Grunt API Lifecycle Management Plugin

- Configuration Management - apigee-config.js

```
exports.xmlconfig = function(env, grunt){
    config = { "test" : [
        {
            "options": {
                "xpath": "//APIProxy/Description",
                "value": "<%= grunt.option('gitRevision') %>"
            },
            "files": {
                "target/apiproxy/<%= apigee_profiles[grunt.option('env')].apiproxy %>.xml": "apiproxy/*.xml"
            }
        },
        {
            "options": {
                "xpath": "//TargetEndpoint/HTTPTargetConnection/URL",
                "value": "https://weather.yahooapis.com/forecastrss"
            },
            "files": {
                "target/apiproxy/targets/default.xml": "apiproxy/targets/default.xml"
            }
        },
        {
            "options": {
                "xpath": "//ProxyEndpoint/HTTPProxyConnection/BasePath",
                "value": "/weathergrunt"
            },
            "files": {
                "target/apiproxy/proxies/default.xml": "apiproxy/proxies/default.xml"
```

Why choose Apigee's Maven?

- ✓ More time to focus on what really matters by automating repetitive tasks
- ✓ Innovation ready. Extensible plugin-based platform
- ✓ Promotes productivity. Promotes usage of CLI (Command-Line Interface). No need for IDEs
- ✓ Easy to adopt. No need of CLI. Eclipse IDE Support through M2E and IntellijIDEA, WebStorm
- ✓ Easy to configure and to track changes. All of its artifacts can live in version control as text files
- ✓ Multilanguage support. One JVM to rule them all (Ruby, Jython, JavaScript, Groovy, Scala) or even Shell scripts
- ✓ Tens of Thousands plugins ready in Maven Central
- ✓ Backed up by Apigee and the open source community

Hands-on

Putting it all together: Continuous Integration

Grunt

Deploy to Apigee

grunt --env=test --username={apigee_edge_email_address} --password={apigee_edge_password} --debug --curl=true --upload-modules

Use apigee gateway and with Yahoo Weather standard Target

https://{org-env}.apigee.net/{api-basepath}/apigee/forecastrss?w=2502265

Use apigee gateway calling Yahoo Weather through Apigee Node.js as Target

https://{org-env}.apigee.net/{api-basepath}/apigee/forecastweather_node?w=2502265

Let's try it! http://goo.gl/lSZrth

Testing

Testing

Grunt Tests with Mocha and friends http://goo.gl/xHI zSh

- TDD (Test Driven Development) for APIs
- Faster to write than BDD (Behavior Driven Development)
- Mocha Testing Framework, Chai for Assertions


```
describe('Check weather in cities', function() {
     async.each(weatherData.simpleWeatherArray() , function(cityData, callback) {
       it('you should be able to get forecast weather for ' + cityData.name + ' from this API Proxy.', function(done) {
          var options = {
                   url: cityData.url, //'https://testmyapi-test.apigee.net/weathergrunt/apigee/forecastrss?w=2502265',
                   headers: {
                     'User-Agent': 'request'
                   }
          }
           request(options, function (error, response, body) {
               expect(body).to.contain(cityData.name) //Sunnyvale, Madrid
               assert.equal(cityData.responseCode, response.statusCode)
               done()
             })
           })
       callback();
```

Testing

POSTMAN

- Postman an API Testing tool
- Very widely used by Developer and Tester community
- Add Jetpacks - They are awesome $9 upgrades

Testing

POSTMAN with Newman

Newman

- A command-line collection runner for Postman.
- It allows you to effortlessly run and test a Postman collection
- Can be Integrated with your build tools like Maven/Grunt and make it part of your CI build
- https://www.getpostman.com/docs/newman_intro

Testing

JMeter

- Get examples https://github.com/apigee/apigee-deploy-maven-plugin
- Use assertions to
  - Validate response codes
  - Validate payload content
  - Validate schemas (JSON – DRAFT04)
  - Validate response times. Spot network latency and performance issues

Mocks with Nock

- Promotes CDC (Consumer-Driven Contract)
- Promotes faster development
- No backend? No problem
  - Issues with starting development without a contract in place:
    - There's no formality, downstream systems changes, no one knows! ☹
    - Downstream systems can run tests to verify whether they're breaking the contract
- There's a backend? No problem
  - Nock can record request and response objects
    - Request/response: content, headers, status codes, delay, etc.

Demo

Putting it all together: Continuous Integration

POSTMAN/Newman

Let’s try it!

Goto Postman and Create a collection of tests

Download your collection

Install newman: npm install -g newman

Run your collection: newman -c mycollection.json

Run your collection 10 times: newman -c mycollection.json -n 10

Mocha/Chai

Adding Tests

Add to `tests/weatherapi.js`

Adding Data Driven Tests

If your tests need data that can be fetched via XHR, stick a `.json` file in the `data` directory, you can access it at `/data/<filename>.json`.

Let's try it!

http://goo.gl/pDxzYG

Source Code Management

SCM

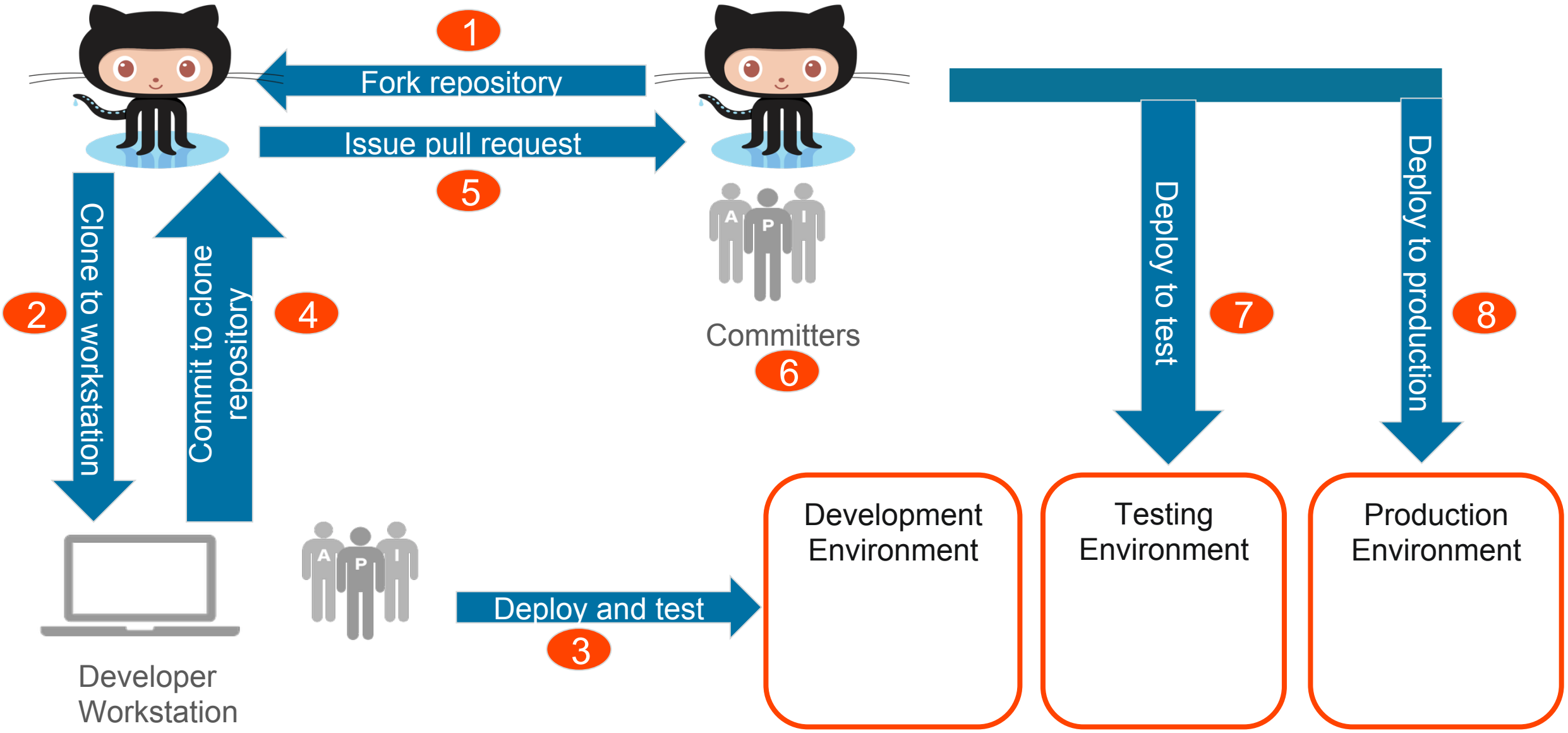
- Define your branching and merging strategy from the get go
- Opt for a scalable model
  - Single Repo vs. Multiple Repos
- Communicate and provide feedback through pull requests (aka. social coding)
- Apply CI and avoid big bang merges
- Practice, practice, practice

SCM

What can be managed in SCM?

- API Artifacts
  - API proxy source code (XML, JS, Java, Python, binaries, etc.)
  - API Documentation (Markdown, HTML)
- Testing Artifacts
  - Scripts
  - Data
- Configuration, Deployment and management scripts
  - Management API requests to create entities like target servers and data stores
  - Configuration Data
- Keep sensitive data out of SCM

Source code management: “Fork-and-pull” model



apigee

Summary

- Identify benefits of SCM
- Apply fork and pull requests
- Learn SCM models pros and cons
  - Monolithic vs Single and Multiple Repos
- Identify SCM branch types and how to use them
- Apply merging

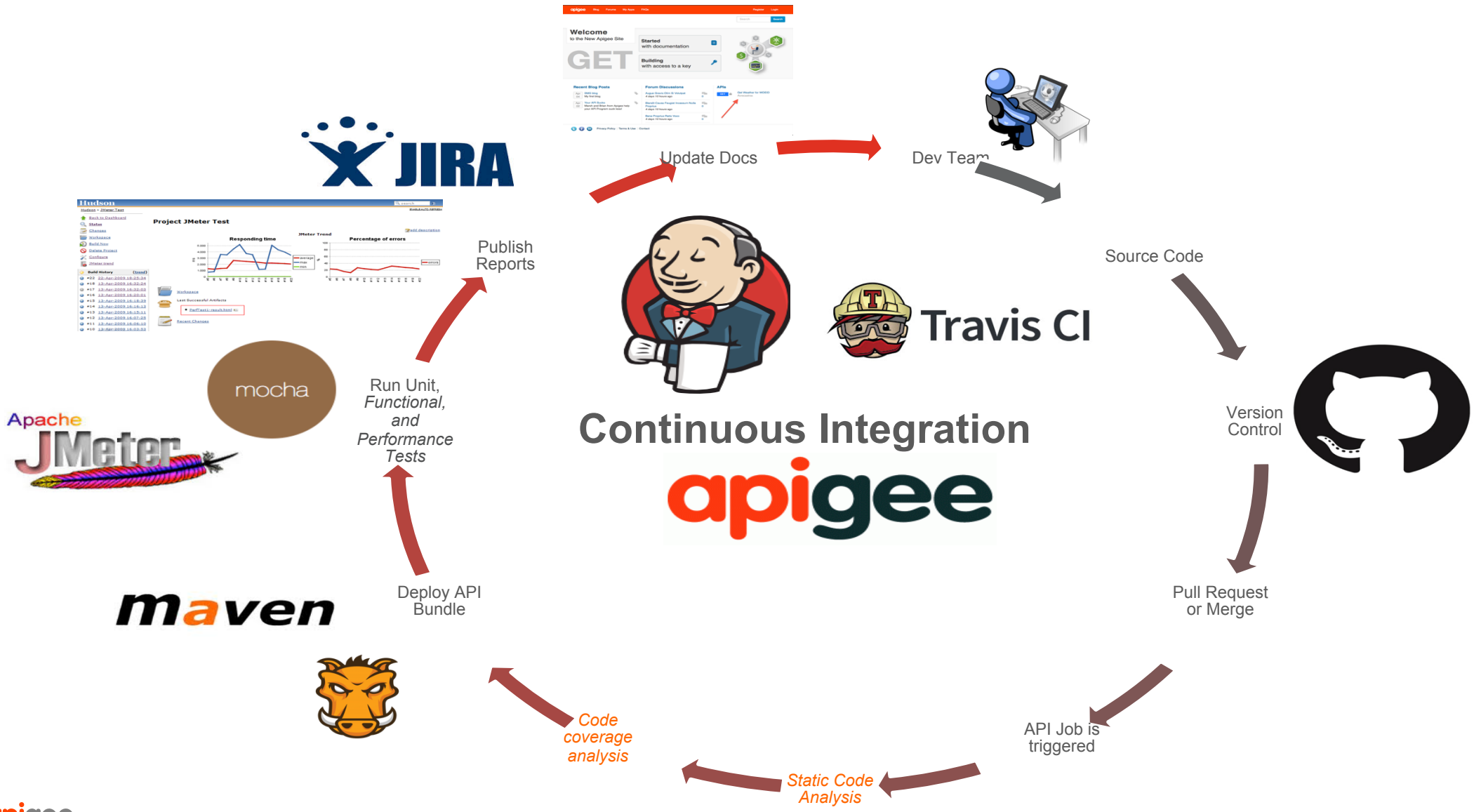
Continuous Integration

Continuous Integration

Principles

- Maintain a code repository
- Automate the build
- Make the build self-testing
- Keep the build fast
- Make it easy to get the latest deliverables
- Everyone can see the results of the latest build
- Automate deployment

The Full Circle



Continuous Integration and Deployment

- API Lifecycle
- Tooling: Jenkins and Travis
- Connect to a Git Repo
- Leverage
  - Maven Plugin
  - Grunt

Reap Your Benefits!!!

- Makes it visible and measurable!!!
- Faster to Market!!!
- Save on Maintenance $$$!!!

Demo

Putting it all together: Continuous Integration

Travis Integration

If you do not have git installed, then

- Fork my repository https://github.com/rakeshtl/CI-Travis.git

If you have git installed, then

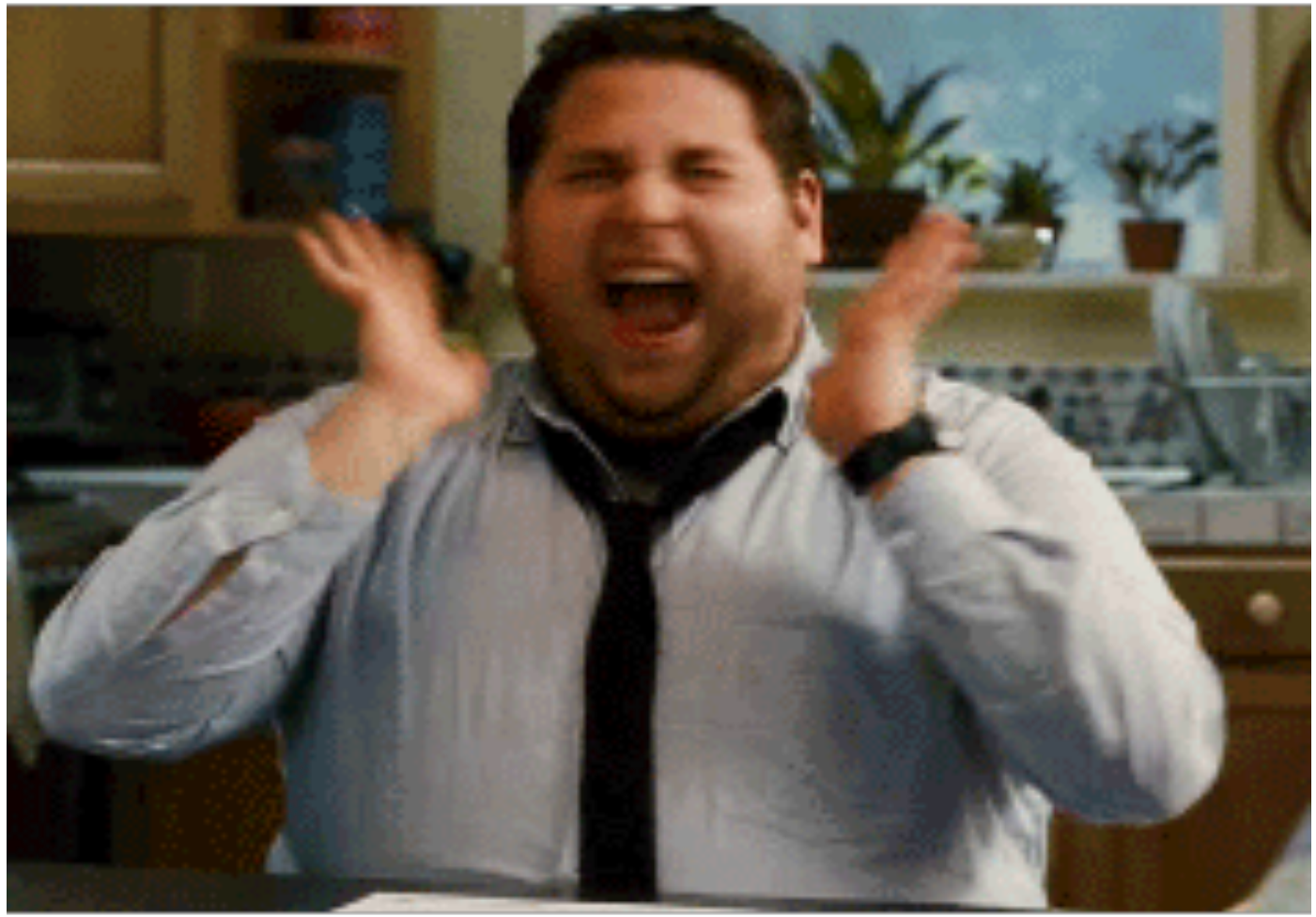
- Create New Public Repository - https://github.com/new
- Goto your directory and run the following commands
- Npm install –g git
- git init
- git add .
- git commit -m "first commit"
- git remote add origin https://github.com/{....} (eg. https://github.com/rakeshtl/grunt-tests.git)
- git push -u origin master

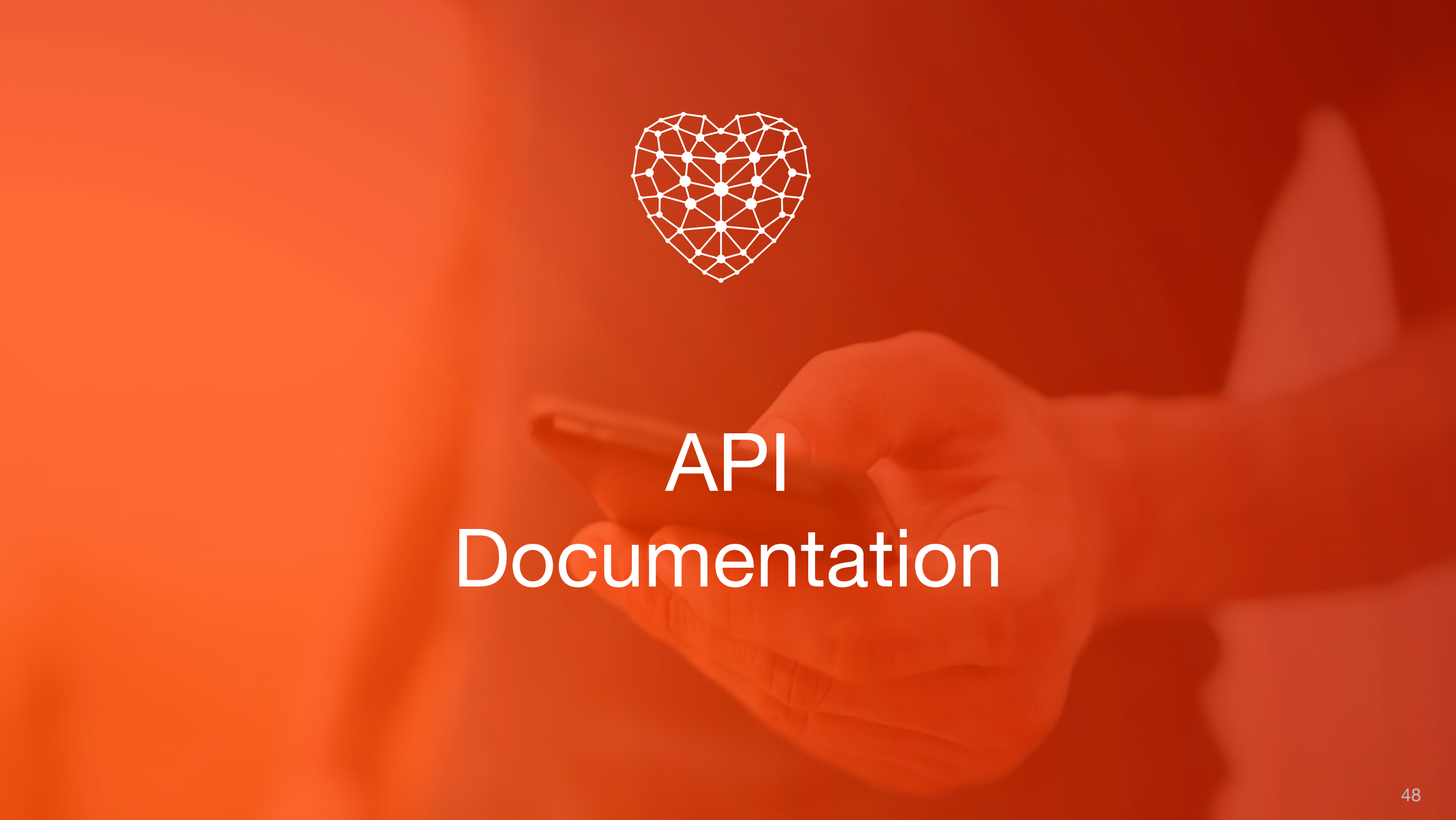
CI on Travis

- Goto https://travis-ci.org
- Add a New Repository
- Goto your Profile, Flick the repository switch on (toggle the checkbox)
- Add the two environment variables - ae_username, ae_password
  - Provide Apigee Credentials
- Ensure .travis.yml file is in your repository
- Trigger your first build with a git push (git add, git commit -m "committing", git push)
- Goto your home page on Travis and watch the build
- Check your Email for Notifications.

Let's try it! https://travis-ci.org

Go For IT!!!





API Documentation

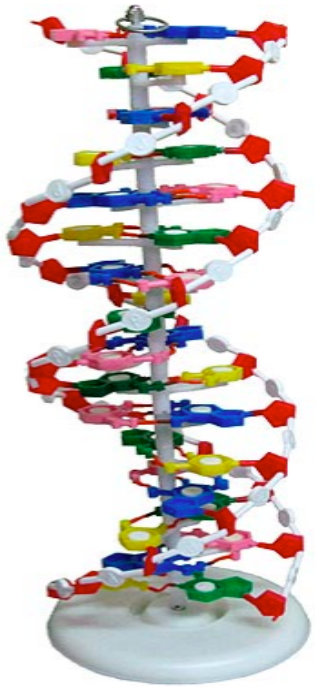
Things to Think About…

- Interactive documentation is becoming the standard for documenting your APIs (e.g. swagger).
- Always treat documentation as code and keep it in version control.
- Functional changes to code likely change how consumers use the API.
- Deploy documentation when you deploy the API code.

Apigee SmartDocs Overview

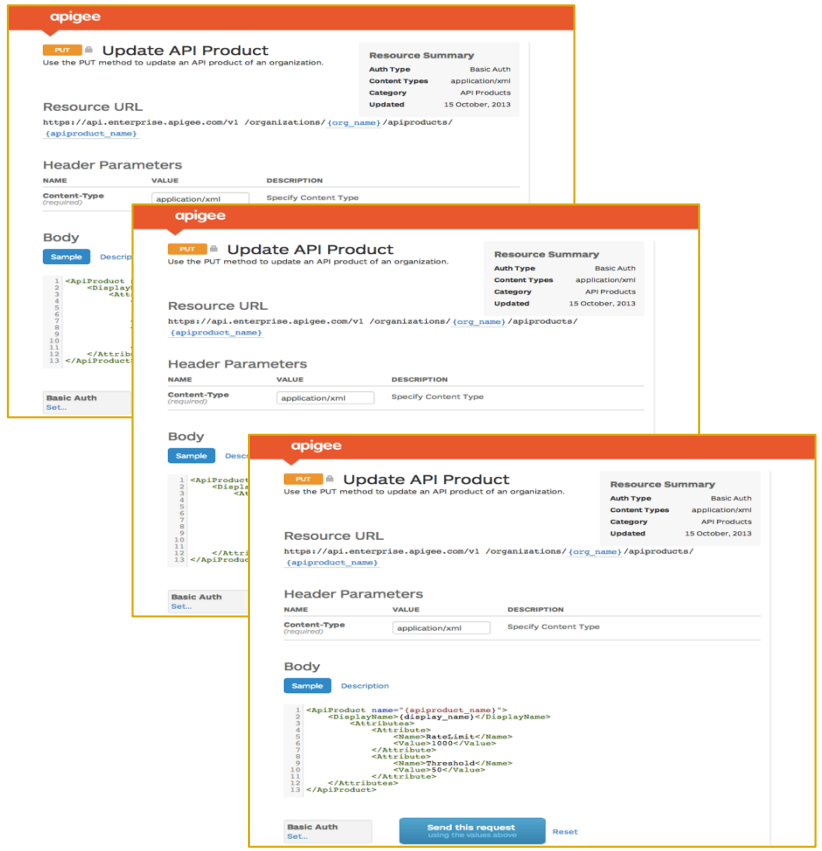
API Modeling

Describe an API structure



SmartDocs

Generate interactive documentation



API-based

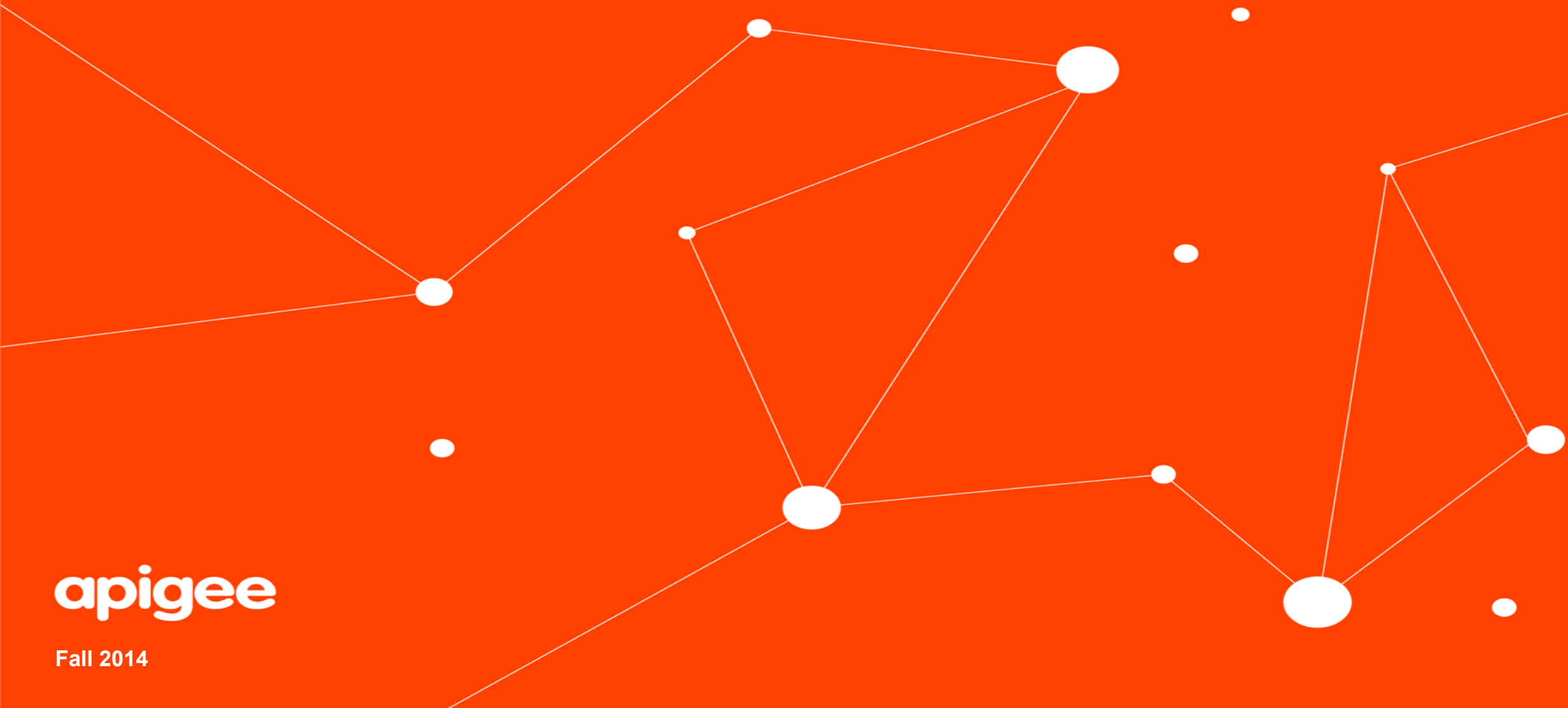
Integrate with any portal / CMS

Apigee Edge Developer Services

Other CMS

gh-pages

Thank you



apigee

Fall 2014