Automation & Governance

agility in an API-first world

Definition of Done

Feature: Publish APIs for our ACME business unit

As a Business Sponsor I want to employ enterprise best practices to build, document, test and publish a secure API that is easy to maintain. I want regular assurance that it is reliable, robust and resilient.

Stage 1: Adoption

- Select an API management platform
- Deploy an secure API management platform
- Securely wire it to your backend services
- Understand your API architecture patterns
    - Internal, B2B, B2C, E2E, public, etc
- Establish an competent API practice

Stage 2: Acceleration

- Automate infrastructure provisioning
- Document a governance model
    - Embrace the open API specification
- Standardise non-functional components
- Automate software development lifecycle
- Adopt micro-services

Stage 3: Governance

- Accurate Documentation
- Re-usable Micro Services
- Re-purpose Legacy Services
- Effortless Interoperability
- Rigorous Quality Assurance
- Monitor Business Continuity

Stage 4: Agility

- Automate API integration
- Automate micro services
- Automate testing
- Automate documentation
- Automate governance

Standardise

Simplify

Celebrate

Standards: Open API

- Defines the features of the API
- The operations available
- The outputs provided
- Data validation
- Error definitions
- Security definitions
- Custom definitions

Simplify: Specifications

```
Communicate with Stakeholders

Interact with Web APIs

Interact with Web Apps

Work with Files

Work with Variables

Work with Templates

Build Software, Sites, Documents
```

Business Defined Development (BDD)

```
Scenario: a goal I wish to automate

  GIVEN    some context

  WHEN     an action is performed

  THEN     an outcome is expected
```

Documentation

Scenario: Generate API documentation

As an **API Program Manager** I want API documentation to support internal users and partners.

Given I am creating documentation

When I read context from project.json

And I read openapi from swagger.yaml

And I build some openapi-docs

Then folder ./build/openapi-docs should exist

Test Cases

```
Scenario: Generate executable test cases

   As a QA Specialist I want to generate
executable BDD test cases for every API.

   Given I am building API test cases

   When I read context from project.json

   And I read openapi from swagger.yaml

   And I build some openapi-tests

   Then folder ./build/openapi-tests should exist
```

Development

Scenario: Generate a NodeJS micro-service

As a **Software Engineer** I want to generate a skeleton micro-service using my favourite framework.

Given I am building a NodeJS micro-service

When I read context from project.json

And I read openapi from swagger.yaml

And I build a micro-node

Then folder ./build/micro-node should exist

Deployment

Scenario: Generate deployable API proxy

As a **Business Manager** I want to deploy an API that complies with enterprise best practice.

Given I am building an API proxy for Apigee Edge

When I read context from project.json

And I read openapi from swagger.yaml

And I build an apigee-proxy

Then folder ./build/apigee-proxy should exist

Assurance

Feature: Automate API assets for ACME

As a Business Sponsor I want to employ enterprise best practices to build, document, test and publish a secure API that is easy to maintain

I want regular assurance that it is reliable, robust and resilient.

Continuous Testing

Executable English

```
Feature: Test HTTP POST
Scenario: Send a dummy JSON payload

Given I am uploading JSON
And I login as robot-tester
And I send dummy-data.json as body
When I POST http://api.acme.example.com/data/
Then response code should be 201
And header Content-Type should exist
And header Content-Type is application/json
```

ApiGeek Architect

```
$ npm install apigeek-architect -g

$ git clone https://github.com/apigeek/example
COMING SOON ...
$ cd example

$ apigeek
```

Executable Example