



SPS COMMERCE



# API DESIGN FIRST SUCCEEDING WITH API GOVERNANCE

APIS FROM DEVELOPERS FOR DEVELOPERS!



## Sponsors

accenture  
<dev>pro

## Community Sponsor

SGI



# TRAVIS GOSELIN

DISTINGUISHED SOFTWARE ENGINEER

DEVELOPER EXPERIENCE

[travisgosselin.com](http://travisgosselin.com)

[linkedin.com/in/travisgosselin](https://linkedin.com/in/travisgosselin)

@travisjgosselin



## SPS COMMERCE

INFINITE RETAIL POWER™

The screenshot displays the SPS Commerce platform's dashboard. At the top, there are four summary cards: 'Pending Partner Acknowledgment' (13), 'New Orders' (218), 'Errors' (21), and 'Ready for Change Acknowledgment' (59). Below these are sections for 'Open Orders' and 'Orders Missing a Shipment'. The 'Open Orders' table lists various trading partners with their order details and status. To the right, a 'RECENT ACTIVITY' sidebar shows a log of events from April 19 and 20, such as new orders and vendor compliance updates.

| Date   | Order #       | Trading Partner       | Status     | Order Amount | Alert |
|--------|---------------|-----------------------|------------|--------------|-------|
| Mar 7  | N05504432-45  | Dick's Sporting Goods | In Process | \$10,677.98  | 1     |
| Mar 14 | 9000-00042618 | Apex Sports           | In Process | \$9,013.27   | 1     |
| Mar 21 | 00007-655543  | Walmart               | In Process | \$22,996.71  |       |
| Mar 23 | P00131866     | Cabela's              | In Process | \$11,901.01  | 1     |
| Mar 29 | G00009114     | Champ's Sports        | In Process | \$971.83     |       |
| Mar 30 | O103064198    | Finish Line           | In Process | \$1,865.31   |       |
| Mar 30 | Q101-044-45   | Foot Locker           | In Process | \$2,113.79   | 1     |
| Mar 31 | 00001-77614   | Gander Mountain       | In Process | \$5,498.23   |       |
| Apr 5  | B005-88760004 | Bass Pro Shops        | New        | \$5,119.66   |       |
| Apr 11 | 0000119-887   | Mills Fleet Farm      | New        | \$782.19     |       |

"

Developer Experience is the activity of studying, improving and optimizing how developers get their work done.

"

theappslab.com (2017)

# DEVELOPER EXPERIENCE

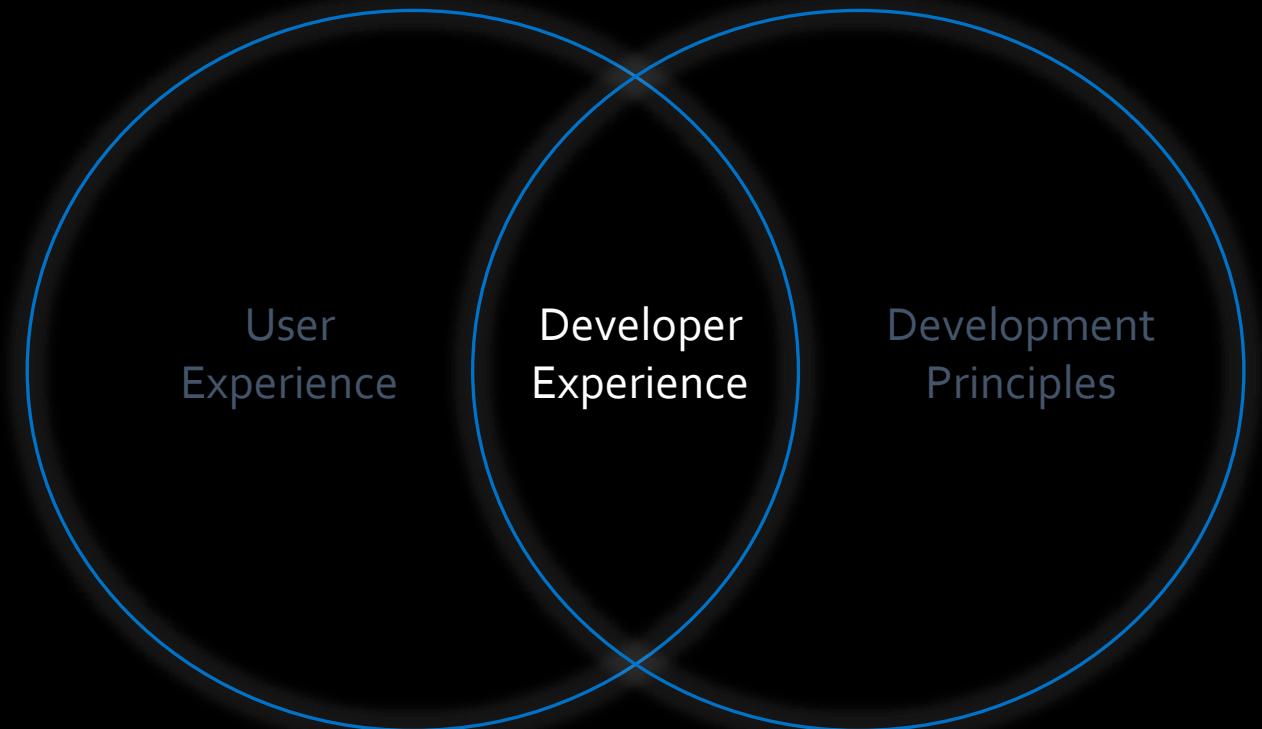
WHAT IS THAT...EXACTLY?

"

Developers work in rainforests, not planned gardens.

"

[a16z.com](http://a16z.com)



# DEVELOPER EXPERIENCE == APIs

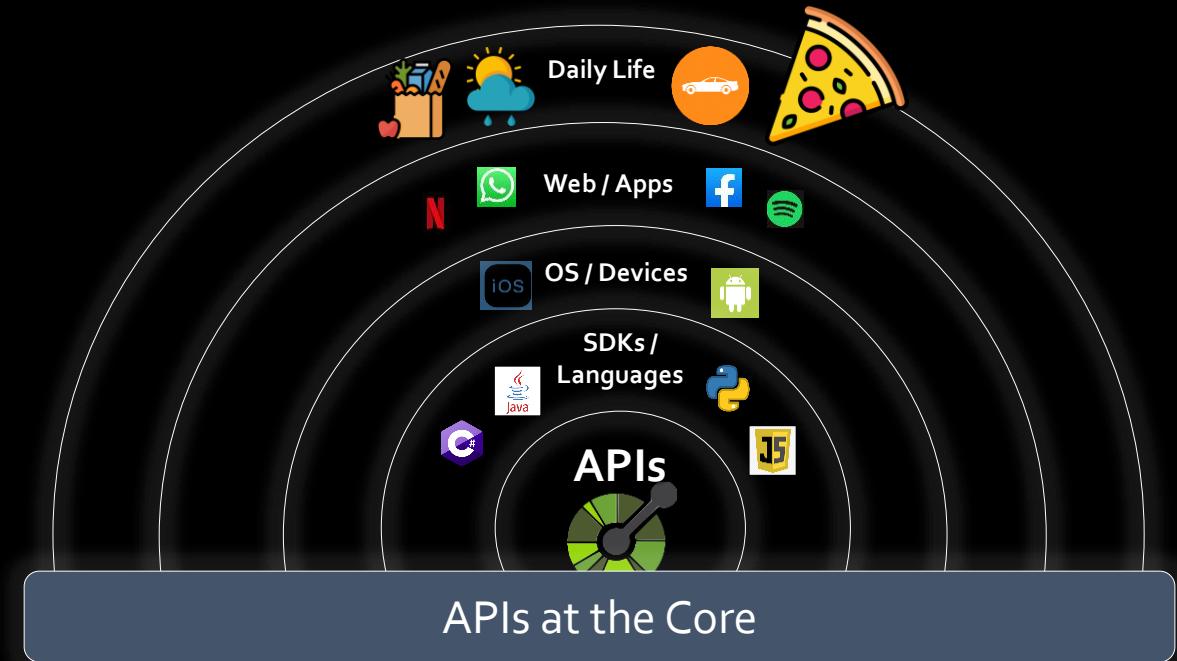
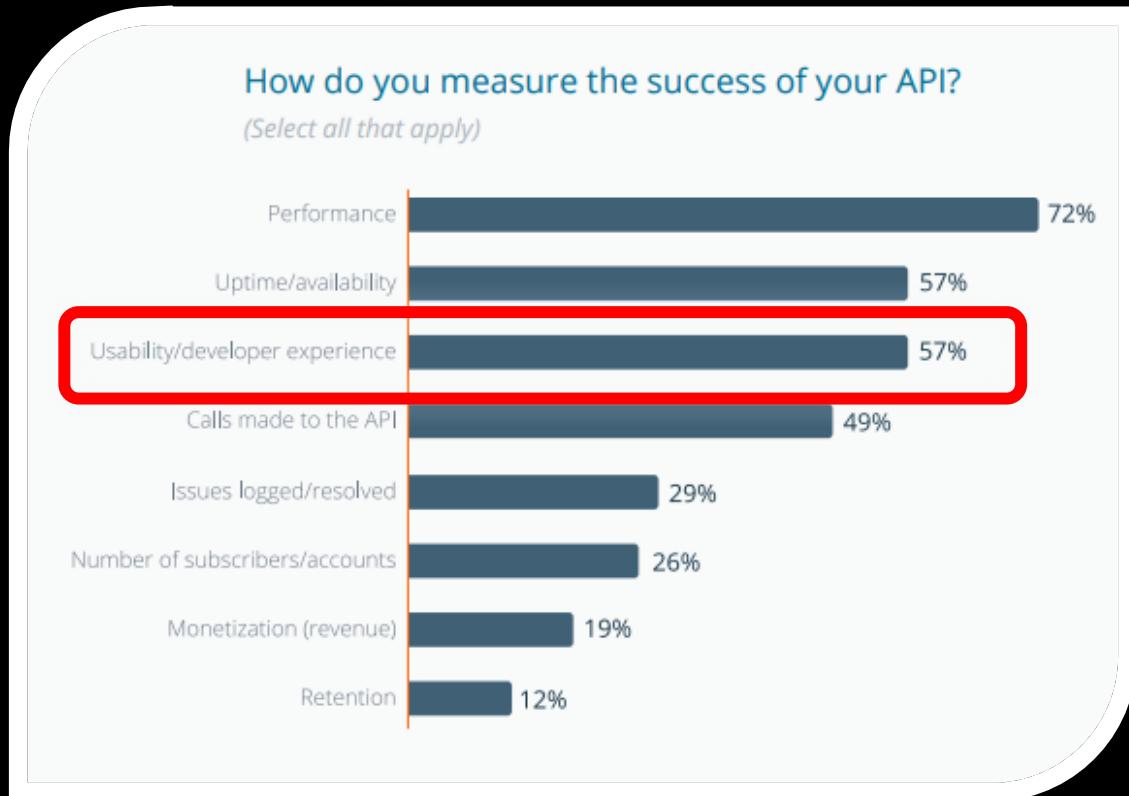
APIs are the Universal Language of Development

Over 90% of Developers Use APIs

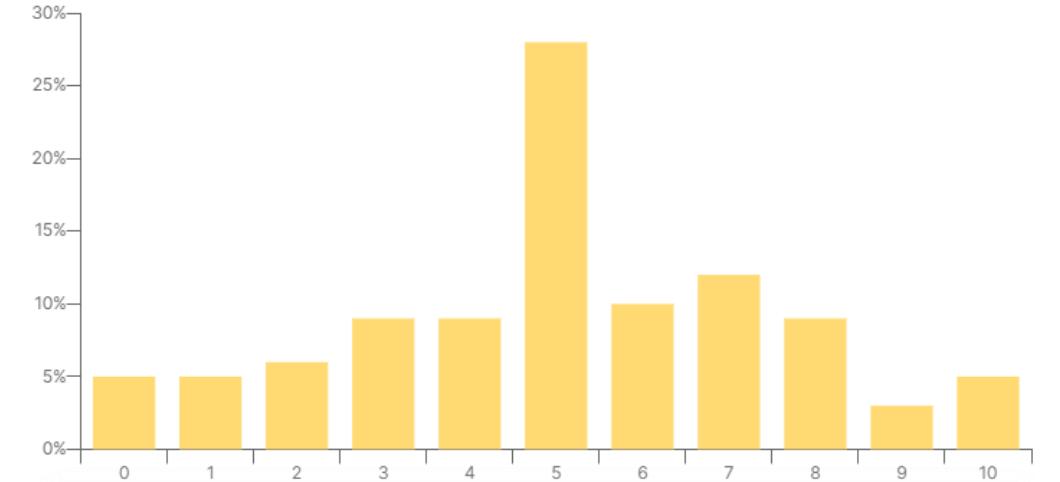
2020 NordicAPI

Developers Spend 30% of Their Time Coding APIs

Second to Manual Debugging: 17%



Embracing API-first



84% of API-First Developers Are Happier!

# API-FIRST APPROACH

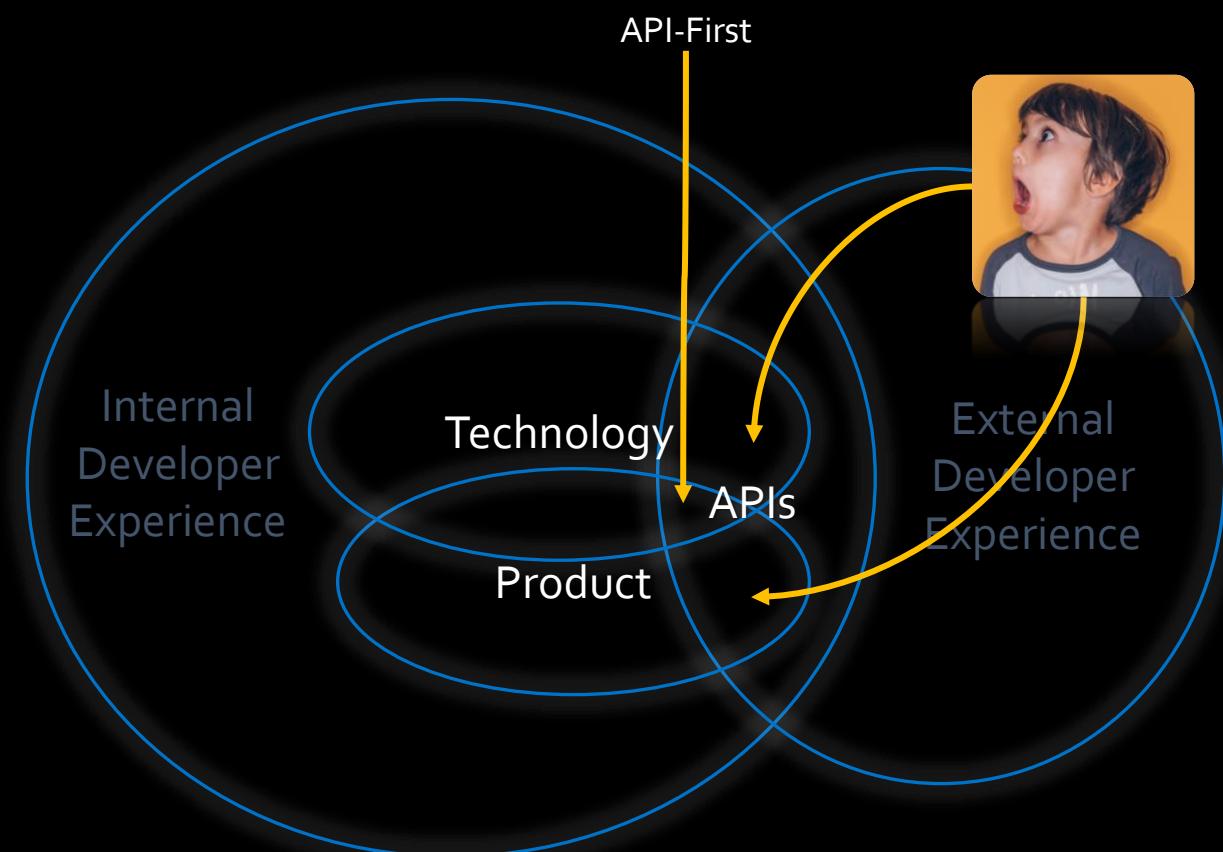
I BUILD APIs... SAME THING?

“

An API-first approach means that [...] your APIs are treated as “first-class citizens.” That everything about a project revolves around the idea that the end product will be consumed [...] by client applications.

swagger.io

”



"

When a design-first approach is not followed, the development process can become chaotic and disorganized. These projects often experience lapses and disconnects between the developers, security, governance, and documentation teams.

[...] It's like being brought in as a building inspector for a house that was built with no blueprints.

"



The API Roadmap from Stoplight (2022)



DESIGNING  
YOUR API

# API GOVERNANCE

Natural Tension in the Enterprise

What we *feel* we want



Rapid Evolution

No Breaking Changes

Velocity

Developer Experience

Existing APIs

Guidelines / Consistency

What we *feel* we need



**autobahn**

# API GOVERNANCE

Natural Tension in the Enterprise

What we *actually* need:

Modern Roundabout

- Specific design and traffic control features
  - Yield control of entering traffic
  - Priority given to circulating traffic
  - Appropriate curvature to ensure low travel speeds
- Designed to create free-flowing, low-speed environment
  - Increases intersection efficiency
  - Eliminates the need for storage lanes
- Safety is enhanced by the elimination of conflict points
  - Possibility for head on collisions drastically reduced
  - 90% reduction in deaths compared to signal controlled intersections



# API GOVERNANCE

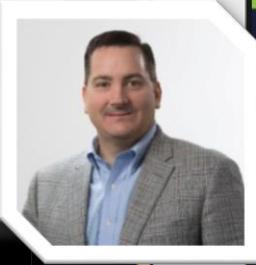
MODERNIZING API GOVERNANCE  
THROUGH API STEWARDSHIP

"

**Stewardship:** The careful and responsible management of something entrusted to one's care.



"



Grow an orchard, don't inspect the fruit.

<https://www.youtube.com/watch?v=sboH7zyfEnA>

A screenshot of a YouTube video player. The video is titled "Keynote: Building APIs at Scale: Moving from API Governance to API Stewardship" by "Mike Kistler &amp; Mark Weitzel, Microsoft". The video has been viewed 0:01 / 28:14. The background of the video player features a green banner with various icons related to APIs and technology. The URL https://www.youtube.com/watch?v=sboH7zyfEnA is visible at the top of the video player.

# API DESIGN MEETS GOVERNANCE

Building API Governance into the API Lifecycle Components

API First

Roadmap

Change Strategy

Taxonomy

Domains

API  
Standards

API  
Design

API  
Development

API  
Publishing

Scaling API Development in the Enterprise

W

## STANDARDS

: a level of quality or attainment.



//

Google.com

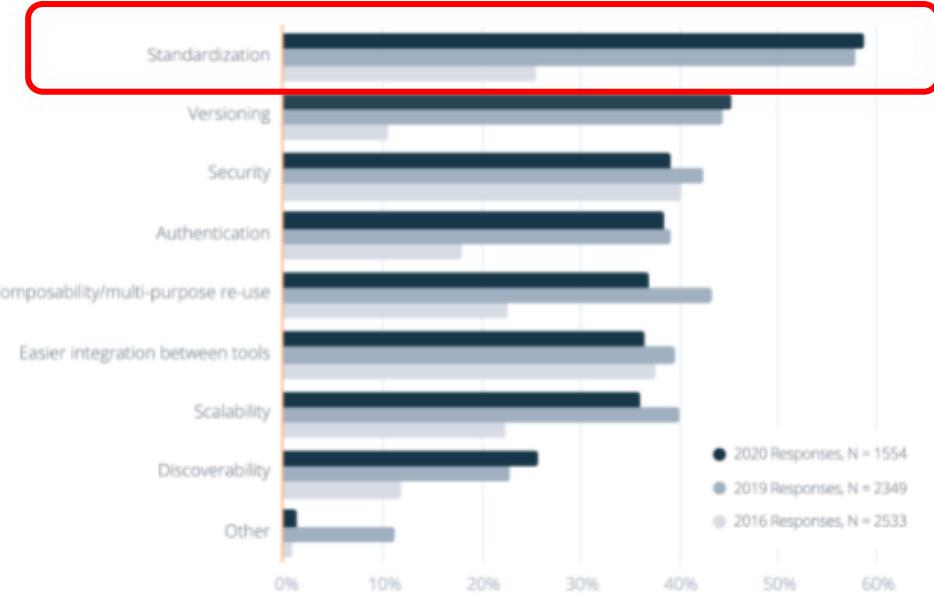
API Standards

API Design

API Development

API Publishing

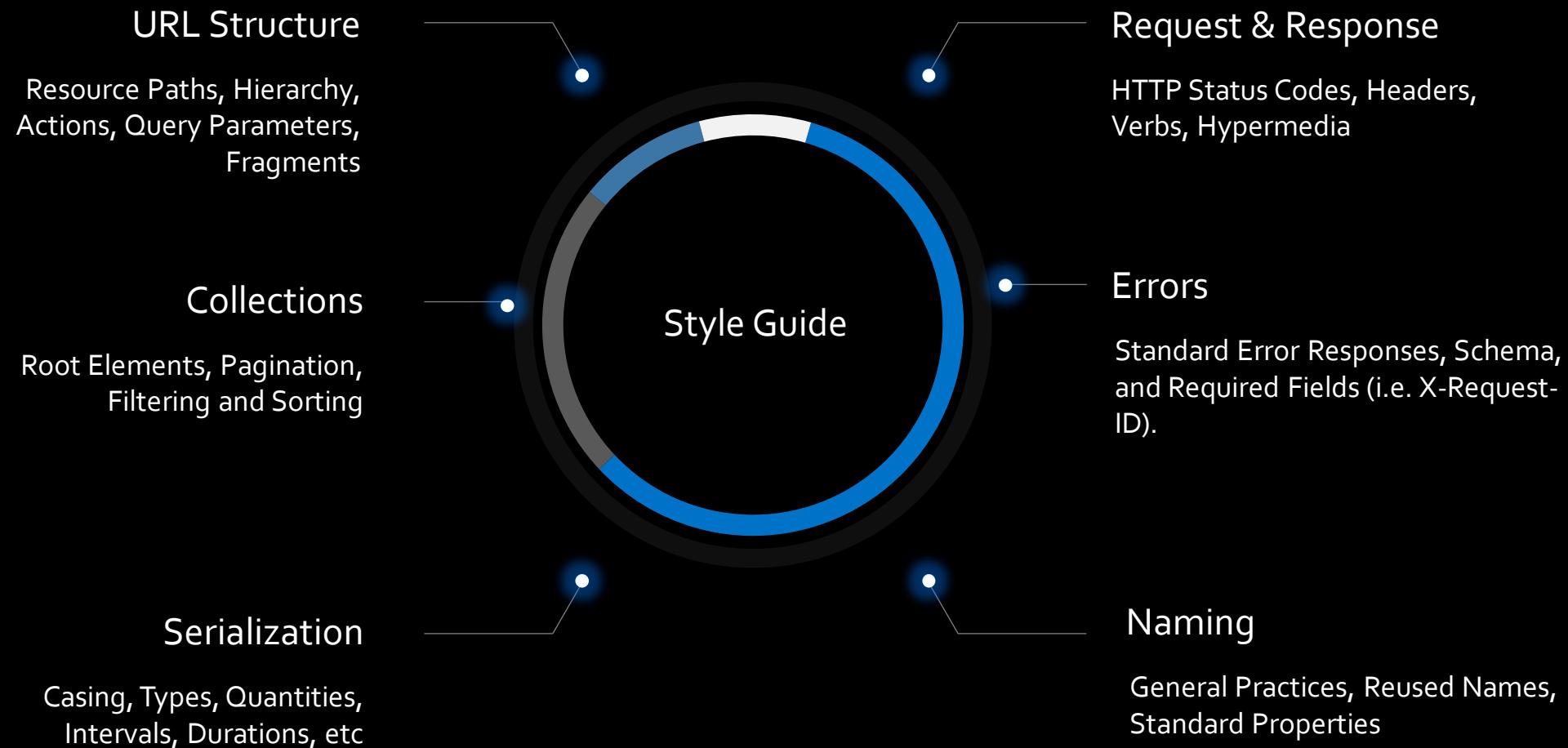
## API Challenges & Future Growth



Standardization continues to be the top API technology challenge teams want to see solved in the coming years

to see solved in the Smart Bear 2020 State of API Report

Style guides inform our decisions when designing API contracts & interfaces.



# API STYLE GUIDE

## Best Practices



### Less Is More

Keep it brief; not a book; not an implementation guide



### Living Document

The guide will mature over time with real-world use cases.



### Goldilocks Use Cases

Beneficial to not over-burden certain standards.



### Align to Existing Standards

Use HTTP and RESTful primitives as intended; use other well-known RFCs!



### Examples ALWAYS!

Inline; post; summary... examples for all! INCORRECT EXAMPLES!

- Bullet points
- Clear statements
- Avoiding statements suggesting implementation

Awareness of implementation implications will direct your style guide for quicker and wider adoption!

**MUST:** means that the definition is an absolute requirement of the specification.

**SHOULD:** means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.

<https://www.ietf.org/rfc/rfc2119.txt>

### HTTP/1.1 403 Forbidden

Content-Type: application/problem+json

```
{  
  "type": "https://example.com/out-of-credit",  
  "title": "You do not have enough credit.",  
  "detail": "Your current balance is 30, but that costs 50.",  
  "instance": "/account/12345msgs/abc",  
}
```

### PROBLEM DETAILS

# APISTYLE GUIDE

## Examples

api-standards/api-style-guide.md x +

github.com/paypal/api-  
2474 lines (1779 sloc)

## HTTP Methods

### Data Resources

### Business Capabilities

Various business capabilities are duplicated across APIs; for example, use-cases.

### HTTP Methods

Most services will fall easily into the acronym CRUD (Create, Read, Update, Delete).

| HTTP Method | Description   |
|-------------|---|
| GET         | To retrieve information from a resource.                        |
| POST        | To create a new resource.                                       |
| PUT         | To update or replace all of the current contents of a resource. |
| DELETE      | To delete an existing resource.                                 |
| PATCH       | Apply a partial update to a resource.                           |
| OPTIONS     | Get information about the options available for a resource.     |

api-guidelines/Guidelines.md at master · Microsoft/api-guidelines · GitHub

github.com/Microsoft/api-guidelines  
2159 lines (1641 sloc)

### 7.4 Supported methods

Operations MUST use the methods listed in this section. These methods are frequently referred to as the standard methods. This specification uses the term "method" to refer to the standard methods.

Below is a list of methods supported by this specification for resources using the methods listed in this section.

| Method  | Description   |
|---------|---|
| GET     | Return the current state of a resource.                                 |
| PUT     | Replace all of the current contents of a resource.                      |
| DELETE  | Delete an existing resource.  |
| POST    | Create a new resource.  |
| HEAD    | Return metadata about a resource without returning the resource itself. |
| PATCH   | Apply a partial update to a resource.                                   |
| OPTIONS | Get information about the options available for a resource.             |

Table 1

#### 7.4.1 POST

POST operations SHOULD be explicitly named, via the method name.

api-design-guide/README.md at master · CiscoDevNet/api-design-guide · GitHub

github.com/CiscoDevNet/api-design-guide/blob/master/README.md

1141 lines (747 sloc) | 56.3 KB

## 3.6 HTTP Verbs

### 3.6.1 POST

3.6.1.1 A POST operation creates a new resource within the collection entity identified by the URL specified in the request.

3.6.1.2 A POST operation creates a new resource within the collection entity identified by the URL specified in the request.

3.6.1.3 A POST operation creates a new resource within the collection entity identified by the URL specified in the request. This URL MUST be unique within the collection entity. Such a request MUST be accompanied by a unique service-generated identifier, and referring to the newly created resource in a `Location` header in the HTTP response, as well as the new resource itself.

**example request:**

```
POST /files/v3/documents
```

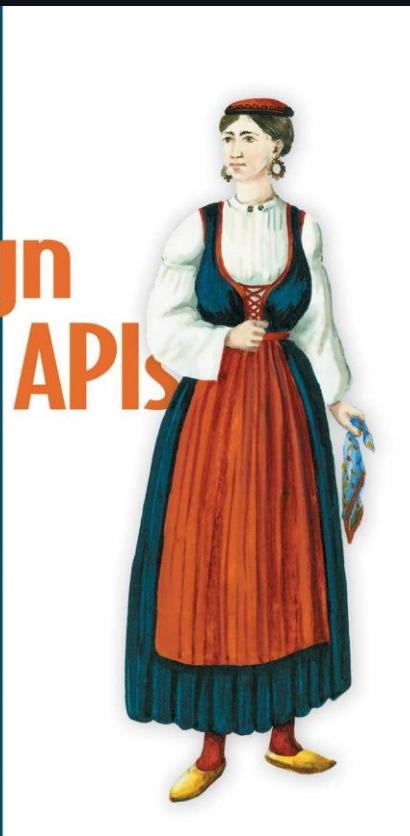
**example response:**

```
HTTP/1.1 201 Created
Location: http://widgets.example.com/files/v3/documents/35bd3e
Content-Type: application/json
Content-Length: 123
```

The Design of Web APIs

Arnaud Lauret  
Foreword by Kin Lane

MANNING



API Handyman: Arnaud Lauret  
ApiStyleBook.com

# SPS STYLE GUIDE

## SPS COMMERCE REST API STANDARDS

cloud foundry

**Consistency** – understanding one resource informs interacts with the rest.

**Discoverability** – API responses guide users without the need for external documentation.

**Simplicity** – complex user workflows constructed from smaller parts.

**Opinionatedness** – one clear way to do something.

**Tolerance** – contracts and consumers are forgiving as possible without compromising security.

**Automation** – lean towards defendable standards, where not compromising

**Experience** – developer experience is king, and major concerns trump the rest.

<https://github.com/spscommerce/sps-api-standards>

### Error Schema

- All `4xx` & `5xx` series of status codes **MUST** come with a consumable JSON error representation as defined in this error schema.
- The error schema defined here **MUST NOT** be returned for a `2xx` series status code (including with the usage of the `207` multi-status code, which is restricted).
- An error or validation response **MUST** follow the error object schema and **SHOULD** have response `Content-Type` of: `application/problem+json`.
- An error or validation response **MUST** include a `requestId` attribute that is used to correlate requests.

```
// REQUEST
POST /articles HTTP/1.1
User-Agent: api-standards-v1
Content-Type: application/json
{
  "foo": "bar",
  "foo2": null
}

// RESPONSE
HTTP/1.1 403 Forbidden
Content-Type: application/problem+json
{
  "title": "You do not have enough credit.", // REQUIRED (string): Short human-readable title of the error that occurs
  "status": 403, // REQUIRED (number): Number representation of the error and MUST match the status code
  "detail": "Your current balance is 30, but that costs 50.", // OPTIONAL (string): Description or detailed human-readable message about the error
  "instance": "/account/12345msgs/abc", // OPTIONAL (URI): Specific URI/Resource that represents a link to the error
  "type": "https://example.com/probs/out-of-credit", // OPTIONAL (URI): URI to human-readable and actionable info about the error

  "requestId": "979f3d3b-a04a-43d7-b55f-8d5609b48783", // REQUIRED (string): Request ID that correlates original request to response
  "context": [
    {
      "code": "INPUT_NOT_NULL", // OPTIONAL (string): Short, machine-readable, name of the validation error
      "message": "Attribute 'foo' must not be null." // REQUIRED (string): Human-readable details or message specific to the error
    }
  ]
}
```

# API STYLE GUIDE

No One Really Cares About Your Guide!

- They WILL NOT be read it.
- Teams will agree with it and deviate.
- Engineers WILL ask questions to answers they could have just searched for or read (LMGTFY).
- Individuals WILL use interpretations that represent conflicts or contradictions “just to be that guy”.
- Years later complaints will continue for decisions already made.



<https://apihandyman.io/nobody-cares-about-api-design-guidelines/>



# API STYLE GUIDE

## Automation



- Every error response (4xx/5xx) SHOULD use the Problem Details RFC 7807 Format

```
--- .spectral.yml
extends:
  - "spectral:oas"
  - your-other-rules.yml
rules:
  operation-description: off

problem-details-error-response:
  description: "Every error response SHOULD support RFC 7807"
  severity: error
  given: $.paths.[*].responses[?(@property.match(^4|5))].content.*~
  then:
    function: enumeration
    functionOptions:
      values:
        1:1 warning oas3-api-servers OpenAPI "servers" must be present and non-empty array.
        2:6 warning info-contact Info object must have "contact" object. info
        2:6 warning info-description Info "description" must be present and non-empty string. info
        7:9 warning operation-description Operation "description" must be present and non-empty string. paths./v1/users.get
        7:9 warning operation-operationId Operation must have "operationId". paths./v1/users.get
        9:9 warning operation-tag-defined Operation tags must be defined in global tags. paths./v1/users.get.tags[0]
        20:30 error unknown-error-format Every error response SHOULD support RFC 7807. paths./v1/users.get.responses[400].content.application
```

```
spectral lint api.yml --ruleset .spectral.yml
```

```
spectral lint api.yml --ruleset
```

<https://raw.githubusercontent.com/SPSCoMmerce/sps-api-standards/main/sps-api-standards.spectral.yml>

“ I think most people just make the mistake  
that it should be simple to design simple  
things.

In reality, the effort required to design  
something is inversely proportional to the  
simplicity of the result.

Roy T. Fielding



# DESIGN

THIS IS WHY IT IS “API DESIGN FIRST”

API Standards

API Design

API Development

API Publishing



Process



People



Technology

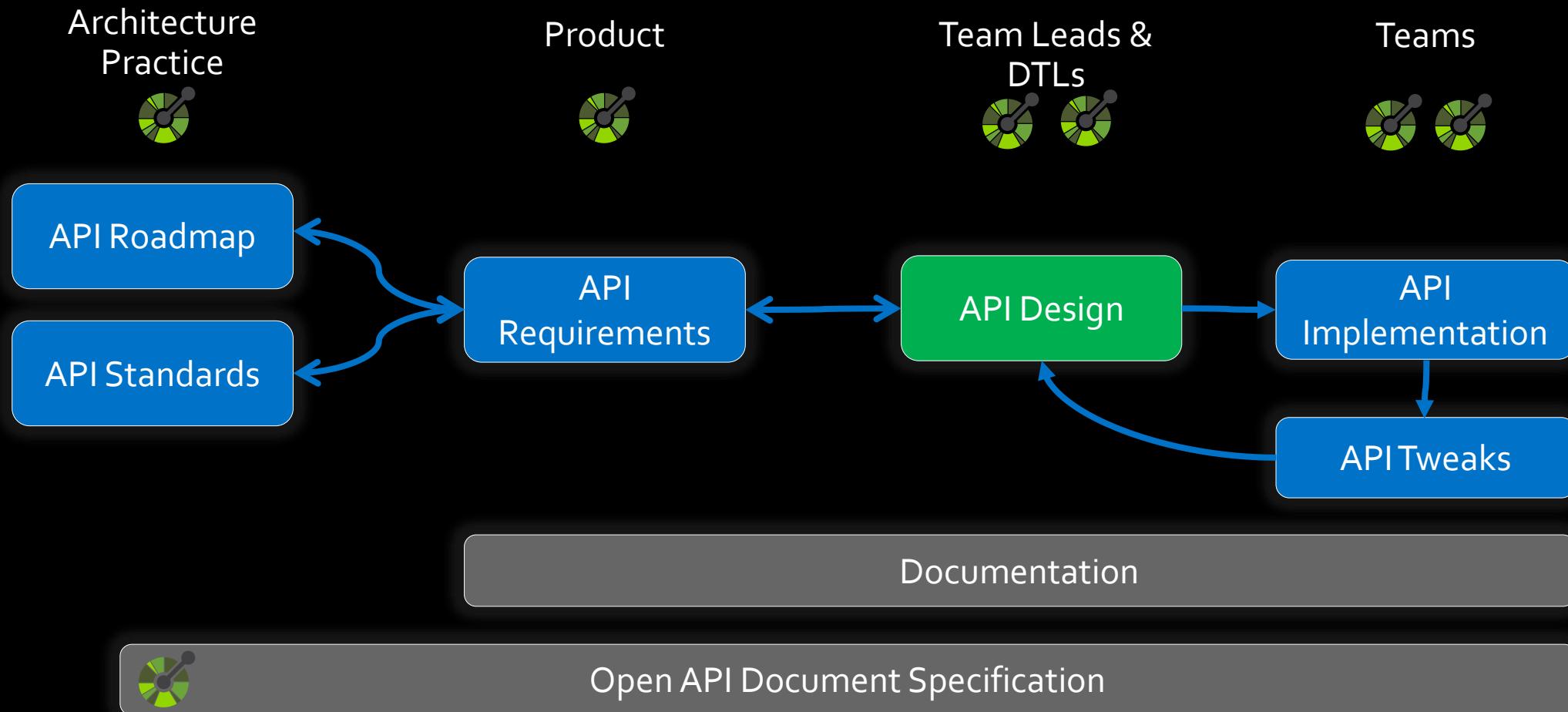
Design Reviews and  
Workflow

Navigating Opinions

Tooling, Automation

# DESIGN

## Organizational Breakdown



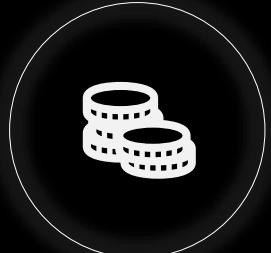
# DESIGN

## Review



### Collaboration

Helping People  
Making it Better  
They Are the Owner



### Transparency

Everyone  
Stakeholders  
Product  
Other Teams



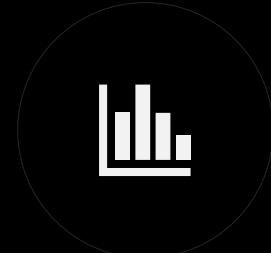
### Consistency

API Standards &  
Models



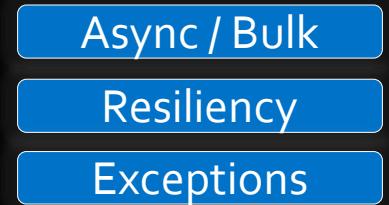
### Strategy & Workflow

Chatty?



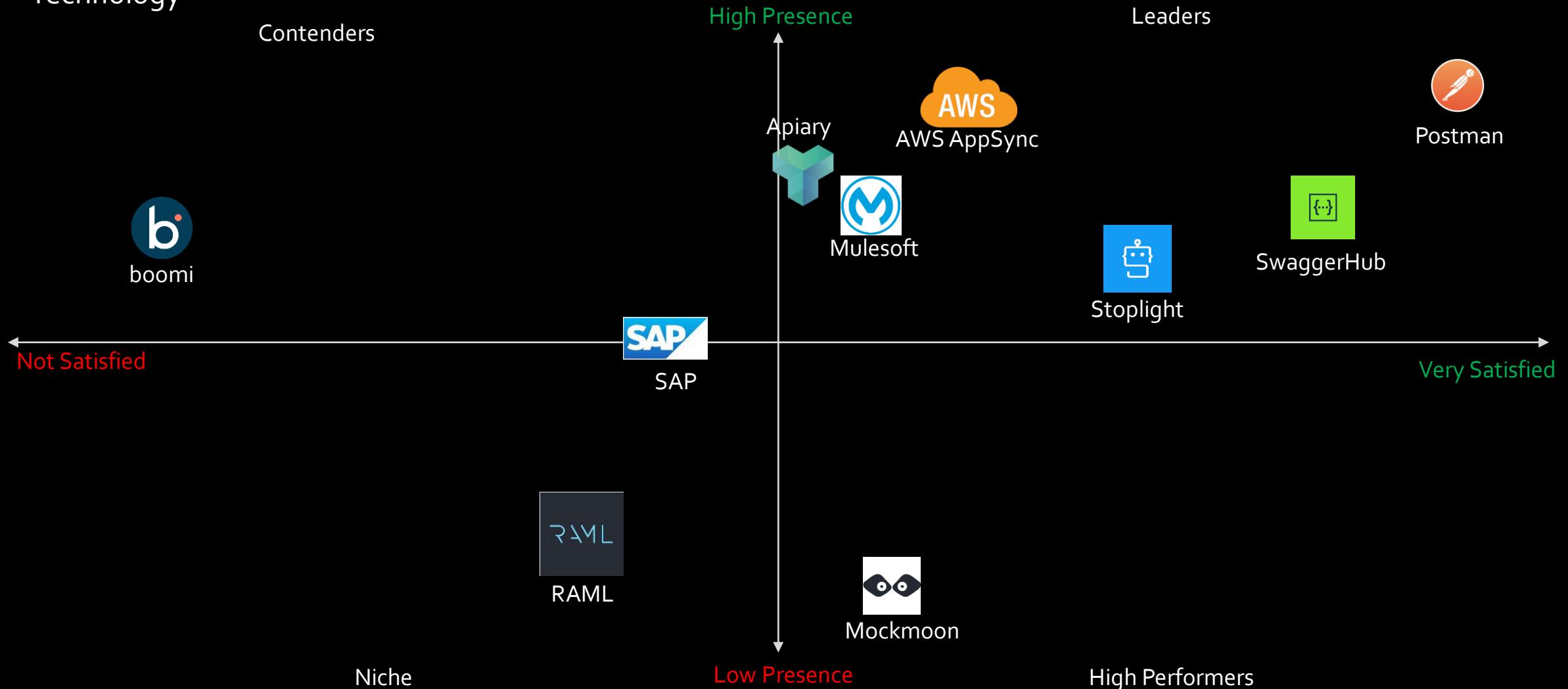
### Learning Opportunities

Everyone is Learning



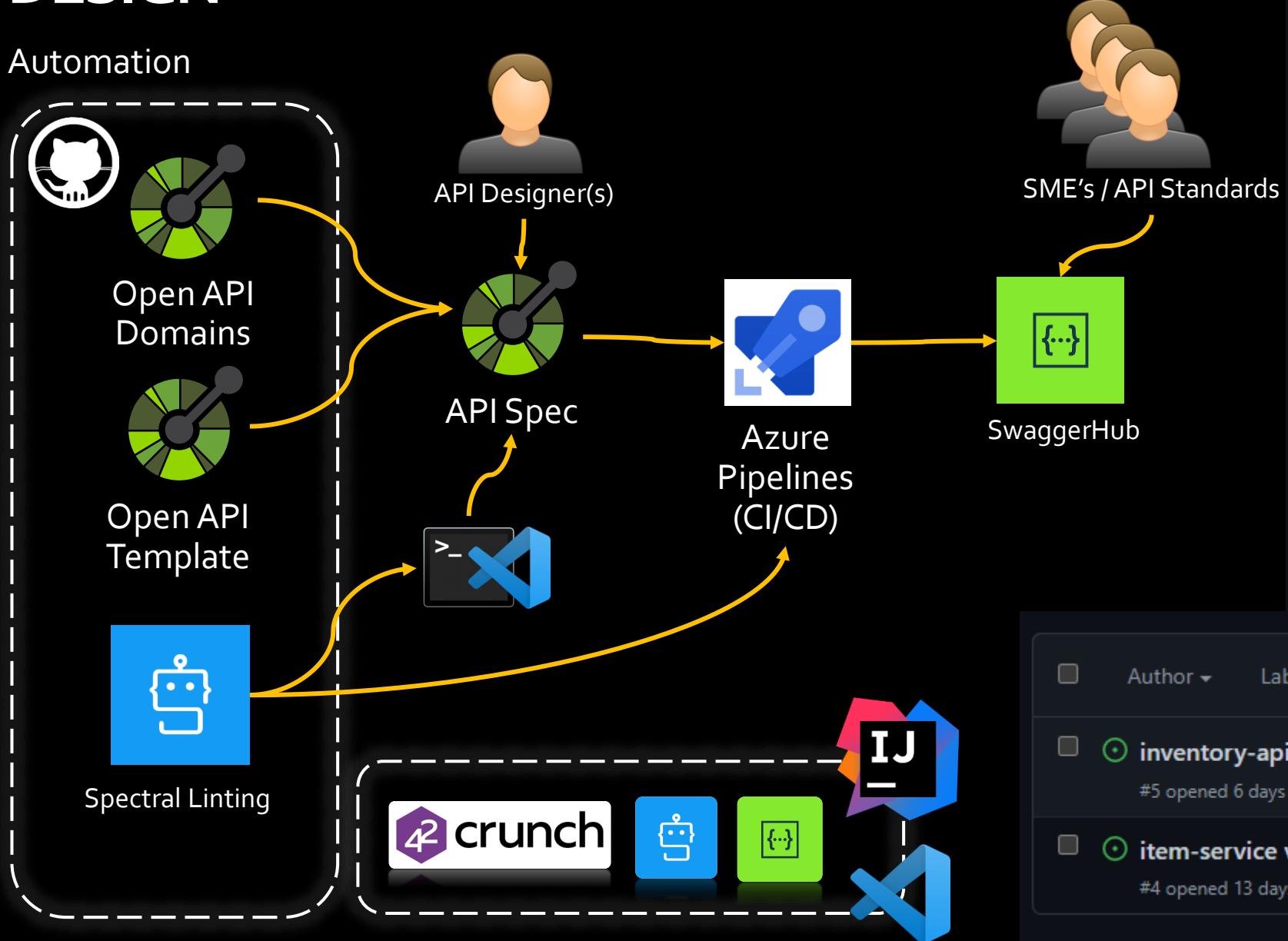
# DESIGN

Technology



# DESIGN

## Automation



SPSCcommerce-VSTS-BOT commented 15 hours ago

### API Validation Results

RESULT: ✓ Your API is Perfect! Well Done! 🎉

API Service ID: 76c5e2e2-99d9-4de8-a45d-8480860b219e  
API Name: azure-pipelines-gates-api  
API Preview URL: SwaggerHub App Designer

### API Standards

```
===== API LINTING RESULTS =====
```

RESULT: 0 Problems (0 errors, 0 warnings, 0 info/hints)

REST API Standards Reference Documentation  
Run Spectral Linting CLI Locally  
Use VSCode/IntelliJ Plugins for Design

### API Changes

```
===== API CHANGE LOG =====
```

Azure Pipelines Gates API

What's Changed

- POST /pipelines/gates/v1/change-management/pipeline Request:  
- Changed application/json  
Schema: Backward compatible

Author Label Projects Milestones Assignee S...

- inventory-api v1: New API api-design-review follow-up new-api  
#5 opened 6 days ago by shifr
- item-service v1: New API api-design-review new-api scheduled  
#4 opened 13 days ago by travisgosselin

Last Saved: 2:30:03 pm - Mar 11, 2022 ✓ VALID

Aa ☀ 8 SAVE SYNC

```
+ 1 openapi: 3.0.2
+ 2 info:
+   title: Inventory API
+   description: Process inventory information
+   version: 0.2.0
+ 6 x-sps-service-id: "34449785-9875-479a-8771
+     -8dd1a99eb0a8"
+ 7 servers:
+ 8   - url: /inventory
+ 9 paths:
+ 10  /v1/items:
+ 11    get:
+ 12      summary: Get Inventory Items
+ 13      description: >-
+ 14        Use this endpoint to Retrieve all
+           Item inventory. Can filter based on
+           a
+ 15        Partner or only get updates that have
+           occurred since a given date.
+ 16        operationId: v1_items_get
+ 17        parameters:
+ 18          - description: >-
+ 19            Filters the list of Item
+              Inventory, returning only
+              inventory
+ 20            information for Items from the
+              specified partner.
+ 21            required: false
+ 22            schema:
+ 23              title: Partnerid
+ 24              type: string
+ 25              description: >-
+ 26                Filters the list of Item
+                  Inventory, returning only
+                  inventory
+ 27                  information for Items from the
+                  specified partner.
+ 28          name: partnerId
+ 29          in: query
+ 30          - description: >-
+ 31
```

## Open Comments

Line 18 | Line 162 | Line 207

tgosselin 4 days ago [Resolve](#) ...  
Should this follow the practice of indicating the type on it? i.e. quantityUpdatedSinceDateTime  
... should it just be quantityUpdatedDateTime?  
or moving towards more standard terminology... quantityModifiedDateTime

This comment has 0 replies.  
[Reply](#)

Line 229 | Line 250 | Line 285 | Line 302 | Line 352

## Resolved Comments

Line 1

# Inventory API

0.2.0 OAS3 SPS COMMERCE

Process inventory information

Note: "Try it out" is disabled because no servers are specified in the "servers" array.  
Please see: [info on OAS3 servers](#)

Servers [/inventory](#)

## default

GET /v1/items Get Inventory Items

GET /v1/items/{spsItemId} Get Inventory Items by spsItemId|internal

POST /v1/imports Import Inventory Data|internal

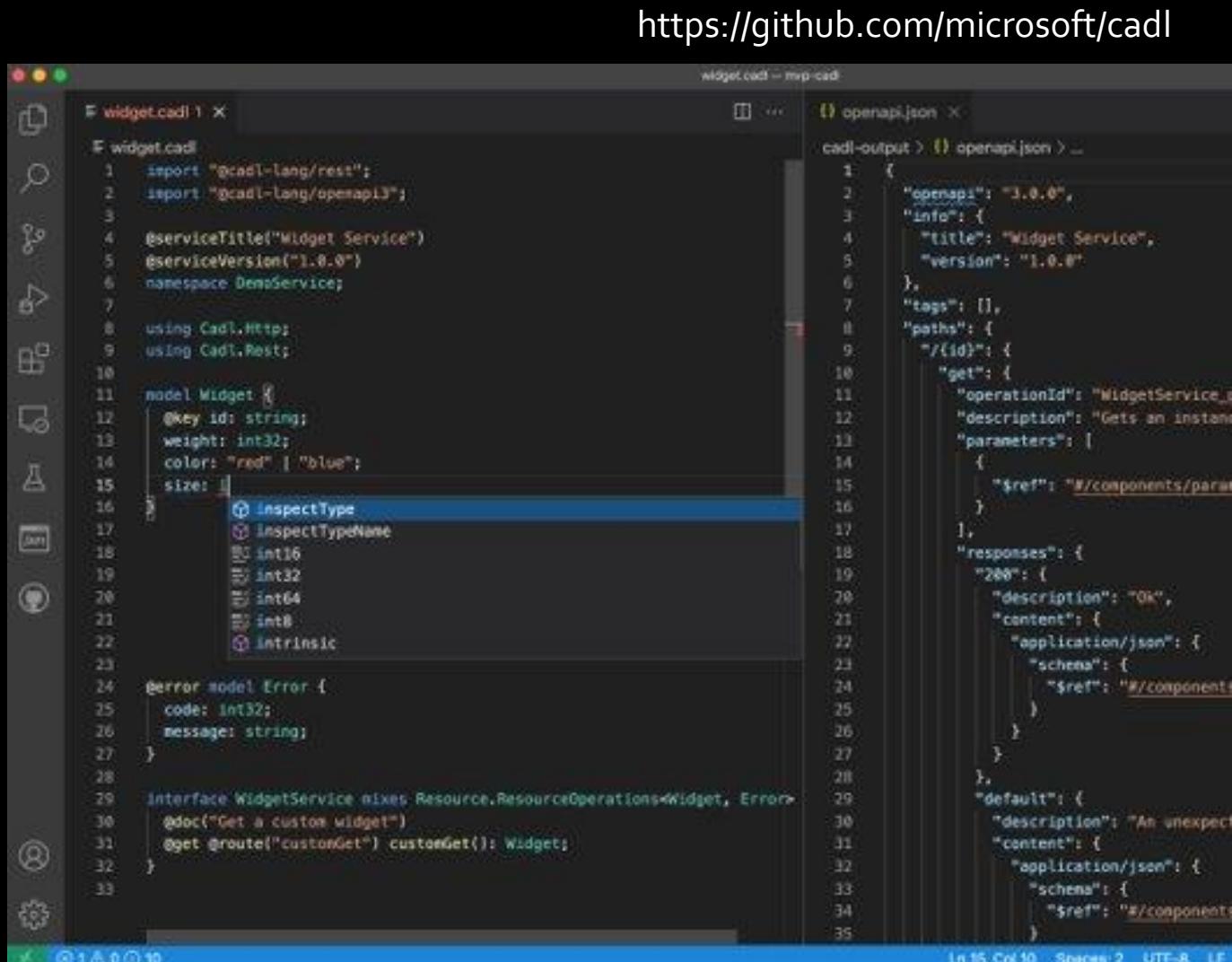
## Schemas

# TypeSpec (Formerly: CADL)

<https://microsoft.github.io/typespec/>

## Automation

- A new language for designing APIs
- Describe API shapes common to REST, GraphQL, gRPC, and others.
- Highly extensible DSL
- Promote reuse through code sharing
- Version Aware
- Generate OpenAPI, Clients, Service Code, Documentation and other assets.
- Fully-integrated VSCode Extension



The screenshot shows two tabs open in VSCode: 'widget.cadl' and 'openapi.json'. The 'widget.cadl' tab contains CADL code defining a 'Widget' model with properties like id, weight, color, and size, and an 'Error' model with code and message. It also includes a service definition with a custom GET method. The 'openapi.json' tab shows the generated OpenAPI specification, which includes the service title, version, and a single endpoint for the custom GET method, mapping back to the CADL code. The interface is dark-themed, and the VSCode status bar at the bottom indicates the file is 15 lines long, column 10, with 2 spaces and LF line endings.

```
widget.cadl (15 lines, 10 columns)  openapi.json (35 lines, 10 columns)

// widget.cadl
1 import "cadl-lang/rest";
2 import "cadl-lang/openapi3";
3
4 @serviceTitle("Widget Service")
5 @serviceVersion("1.0.0")
6 namespace DemoService;
7
8 using Cadl.Http;
9 using Cadl.Rest;
10
11 model Widget {
12     @key id: string;
13     weight: int32;
14     color: "red" | "blue";
15     size: [
16         @inspectType
17         @inspectTypeByName
18         @int16
19         @int32
20         @int64
21         @int8
22         @intrinsic
23
24     error model Error {
25         code: int32;
26         message: string;
27     }
28
29     interface WidgetService混子 Resource.ResourceOperations<Widget, Error>
30     @doc("Get a custom widget")
31     @get @route("customGet") customGet(): Widget;
32 }
33

// openapi.json
1 {
2     "openapi": "3.0.0",
3     "info": {
4         "title": "Widget Service",
5         "version": "1.0.0"
6     },
7     "tags": [],
8     "paths": {
9         "/id": {
10             "get": {
11                 "operationId": "WidgetService_customGet",
12                 "description": "Gets an instance of a widget by ID",
13                 "parameters": [
14                     {
15                         "$ref": "#/components/parameters/id"
16                     }
17                 ],
18                 "responses": {
19                     "200": {
20                         "description": "OK",
21                         "content": {
22                             "application/json": {
23                                 "schema": {
24                                     "$ref": "#/components/schemas/Error"
25                                 }
26                             }
27                         }
28                     },
29                     "default": {
30                         "description": "An unexpected error occurred",
31                         "content": {
32                             "application/json": {
33                                 "schema": {
34                                     "$ref": "#/components/schemas/Error"
35                                 }
36                             }
37                         }
38                     }
39                 }
40             }
41         }
42     }
43 }
```



# DEVELOPMENT

THE OLD STOMPING GROUND!

API Standards

API Design

API Development

API Publishing



Contracts



Interoperability



Validation

Mocking, Stubs &  
SDKs

API Gateways, Reuse  
& Lifecycle

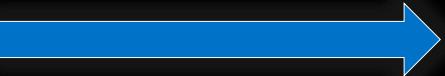
Did you build what  
you designed?

# DEVELOPMENT

## Contracts



```
/v1/users:  
  get:  
    tags:  
      - "Users"  
    responses:  
      "200":  
        description: "Success"  
        content:  
          application/json:  
            schema:  
              type: "array"  
              items:  
                required:  
                  - "id"  
                  - "username"  
                type: "object"  
                properties:  
                  id:  
                    type: "integer"  
                    format: "int32"  
                  firstName:  
                    type: "string"  
                    nullable: true  
                  lastName:  
                    type: "string"  
                    nullable: true
```



## Mocking

```
// REQUEST  
GET https://api.mock.com/my-service/v1/users  
  
// RESPONSE  
[  
  { "id": 23, "firstName": "John", "lastName": "Doe" }  
]
```

Static

Dynamic



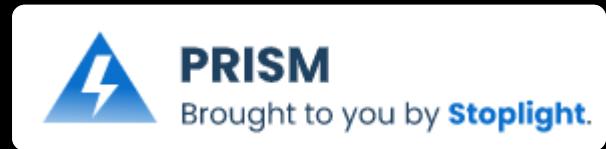
## Stubs

```
[HttpGet]  
[Route("/v1/users")]  
[SwaggerOperation("V1UsersGet")]  
[SwaggerResponse(statusCode: 200, type: typeof(List<InlineResponse200>), description: "Success")]  
public virtual IActionResult V1UsersGet()  
{  
  var exampleJson = "[ {\n    \"firstName\" : \"firstName\", \n    \"lastName\" : \"lastName\" } ]";  
  var example = exampleJson != null  
    ? JsonConvert.DeserializeObject<List<InlineResponse200>>(exampleJson)  
    : default(List<InlineResponse200>); //TODO: Change the data returned  
  return new ObjectResult(example);  
}
```



## SDKs

```
public interface IUsersApi : IApiAccessor  
{  
  List<InlineResponse200> V1UsersGet ();  
  V1UsersBody2 V1UsersIdGet (int? id);  
  void V1UsersIdPut (int? id, UsersIdBody);  
}
```



microsoft/kiota

OpenAPI based HTTP Client code generator



A 46 Contributors    I 111 Issues    D 9 Discussions    S 682 Stars    F 82 Forks

<https://github.com/OpenAPITools/openapi-generator>

# DEVELOPMENT

## Validation

```
=====
          API CHANGE LOG
=====

Swagger Petstore

What's New

- GET /pet/{petId}

What's Deleted

- POST /pet/{petId}

What's Deprecated

- GET /user/logout

What's Changed

- PUT /pet
  Request:
    - Deleted application/xml
    - Changed application/json
      Schema: Backward compatible
  - POST /pet
    Parameter:
      - Add tags in query
    Request:
      - Changed application/xml
        Schema: Backward compatible
      - Changed application/json
        Schema: Backward compatible
      - JSON
```

Design vs Actual (Generated and/or Runtime)

```
docker run openapitools/openapi-diff:latest design.yaml generated.json --fail-on-changed
```

The screenshot shows a CI/CD pipeline interface. On the left, there is a 'Build' step labeled 'V1.1.3'. An arrow points from this step to a 'docker run openapi-diff' step. From the 'docker run openapi-diff' step, another arrow points to a 'Security Scanning Demonstration' step. This scanning step includes a green 'swagger' icon. A red box highlights a warning message: 'API: Open API spec contains errors that DO NOT meet the SPS API Standards (https://docs.platform.spcommerce.com/api-design/standards/). See spectral log output above.' Below the scanning step, there is a 'Warnings 1' section with a single warning. At the bottom, there is a 'Stages' section showing the progression of the pipeline through stages like CONTEXT, COMPILE, INTEGRATION, PROD APPROVAL, and PROD.

Build V1.1.3

docker run openapi-diff

swagger

#4.3.148 Security Scanning Demonstration (#197)

Triggered by travisgosselin

Repositories 2

SPSCcommerce/spsc-ref-dotnetcore-api , +1

Time started and elapsed

Tue at 9:09 PM

1d 14h 24m

Related 0 work items

2 published: 1 consumed

Warnings 1

API: Open API spec contains errors that DO NOT meet the SPS API Standards (https://docs.platform.spcommerce.com/api-design/standards/). See spectral log output above.

INTEGRATION • DOCS\_DEPLOY INTEGRATION DEPLOY • publish documentation

1 approval needs review before this run can continue to PROD APPROVAL

Stages Jobs

CONTEXT 1 job completed 9s 1 artifact

COMPILE 3 jobs completed 3m 40s 100% tests passed 1 artifact

INTEGRATION 4 jobs completed 8m 40s 2 checks passed 1 artifact

PROD APPROVAL Waiting 0/1 checks passed

PROD Not started

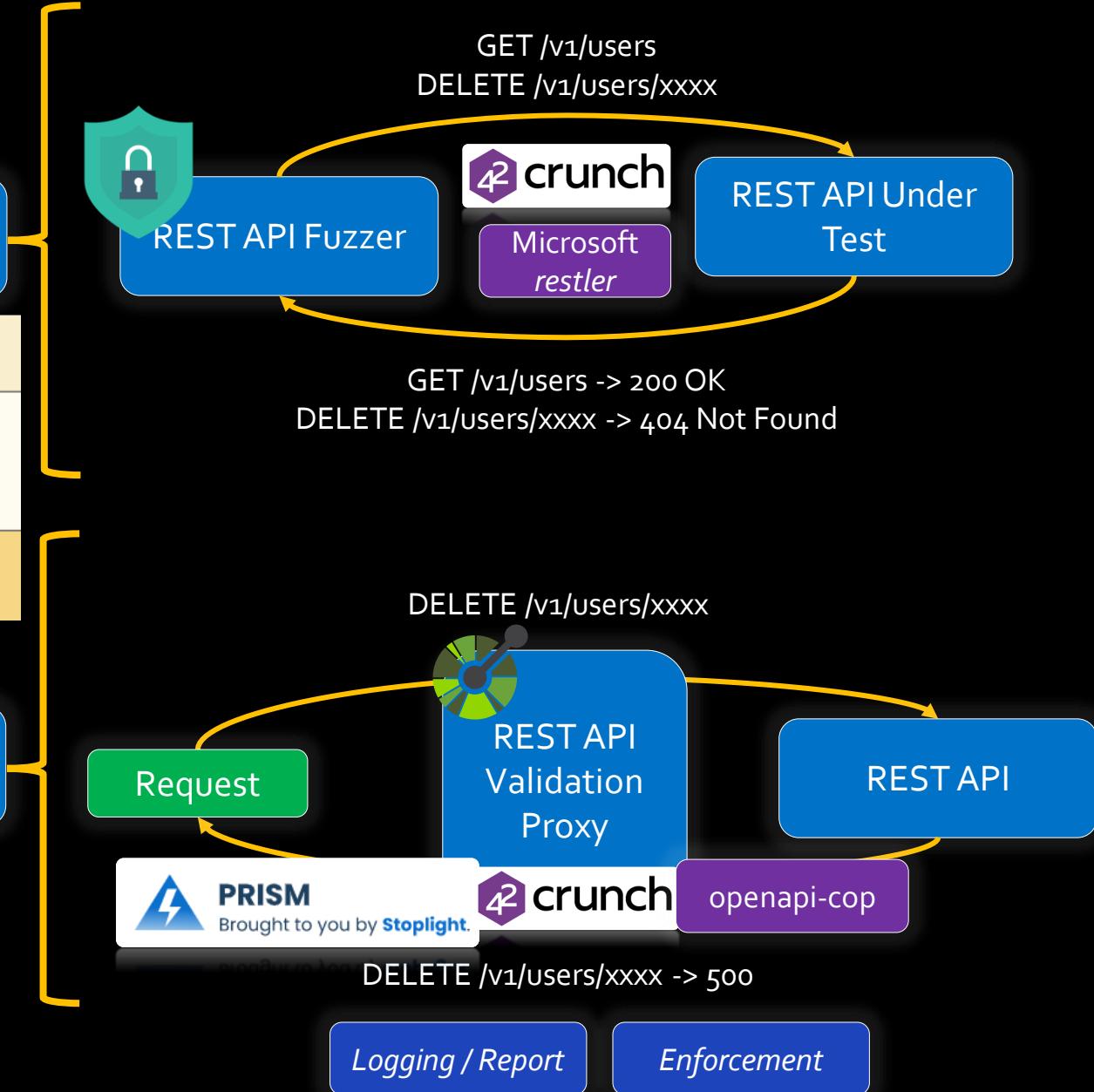
# DEVELOPMENT

## Validation

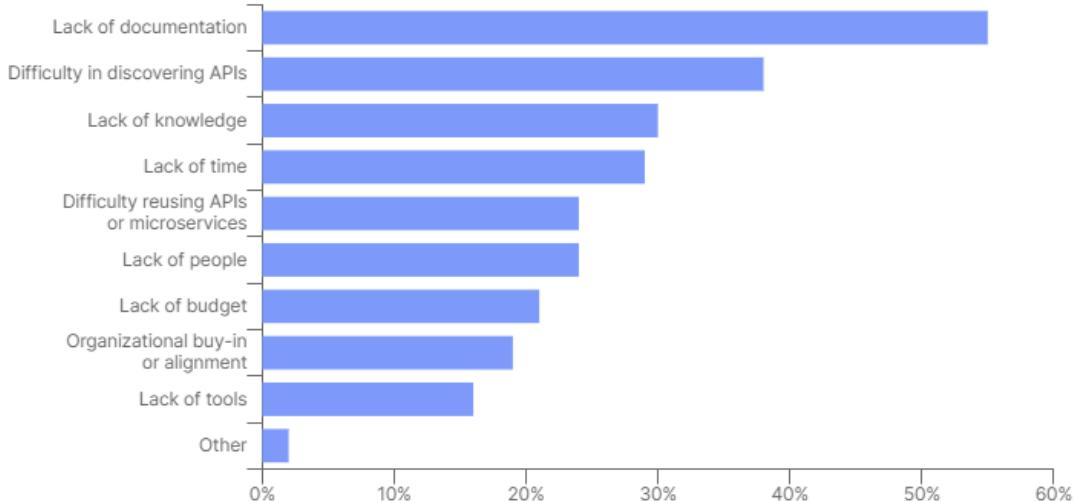


Open API Document

```
/v1/users:  
get:  
  tags:  
  - "Users"  
  responses:  
    "200":  
      description: "Success"  
      content:  
        application/json:  
          schema:  
            type: "array"  
            items:  
              required:  
              - "id"  
              - "username"  
              type: "object"  
              properties:  
                id:  
                  type: "integer"  
                  format: "int32"  
                firstName:  
                  type: "string"  
                  nullable: true  
                lastName:  
                  type: "string"  
                  nullable: true
```



## Obstacles to consuming APIs



API Standards

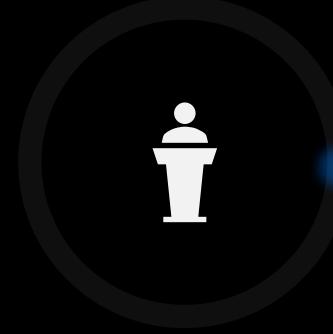
API Design

API Development

API Publishing



API Catalog



Documentation



Organization

# PUBLISHING

## CONSUMERS EXPERIENCE

“ Great API documentation: That's a discipline that's really lacking... ”

James H.

# PUBLISHING

API Catalog



Open API  
Document

Centralized

Eliminate Distributed API Specs,  
Inventory of all APIs

Discoverable

Searchable, APIs & Endpoints,  
Critical for Internal  
Communication

Developer Portal

Internal, External, Partners,  
Interactive, Communication with  
Developers

Documentation

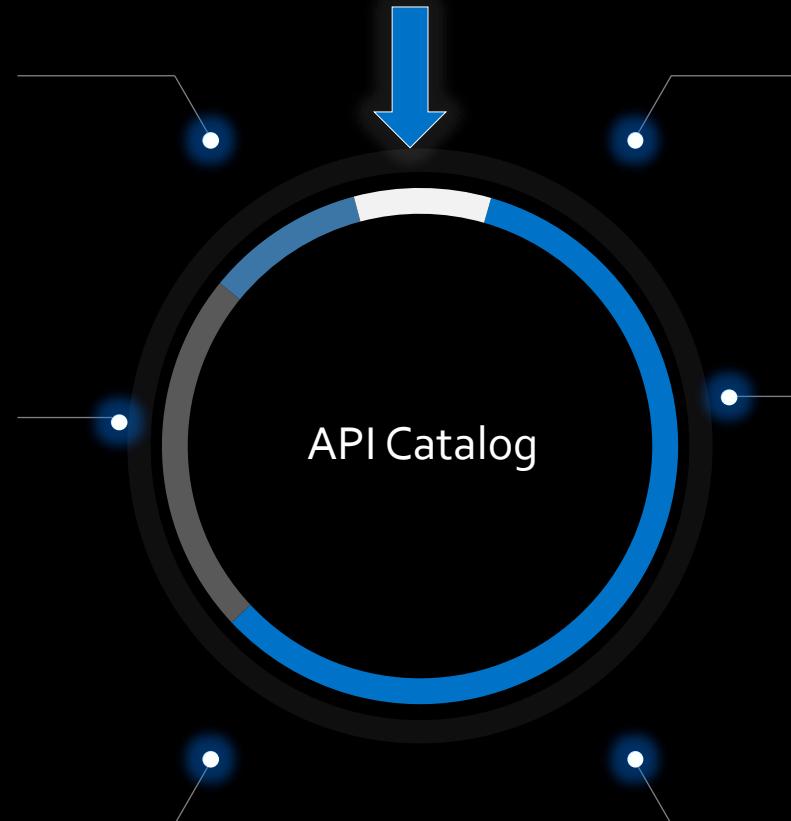
Guides, Tutorials & Examples

Versioned

Change Strategy & Immutable  
Versions Over Time

API

Programmatically Accessible  
through Automation



# PUBLISHING

## Great Documentation Examples

The screenshot shows the Stripe API documentation page for the **Pagination** section. The left sidebar contains a navigation menu with links to various API resources like Balance, Charges, Customers, etc. The **Pagination** section is highlighted with a dark background. The main content area has a dark header with the title **Pagination**. Below the title, there's a paragraph explaining that all top-level API resources support bulk fetches via "list" API methods. It mentions `list charges`, `list customers`, and `list invoices`. These list API methods share a common structure, taking at least three parameters: `limit`, `starting_after`, and `ending_before`. The text then describes how Stripe uses cursor-based pagination with `starting_after` and `ending_before` parameters. It notes that both parameters take an existing object ID value (see below) and return objects in reverse chronological order. The `ending_before` parameter returns objects listed before the named object, while `starting_after` returns objects listed after it. Both parameters are mutually exclusive. A note says that client libraries offer `auto-pagination` helpers to easily traverse all pages of a list. There's also a related video link: [Pagination and auto-pagination](#).

**RESPONSE**

```
{  
  "object": "list",  
  "url": "/v1/customers",  
  "has_more": false,  
  "data": [  
    {  
      "id": "cus_AJ6yEs79rUDTXH",  
      "object": "customer",  
      "address": null,  
      "balance": 0,  
      "created": 1489794306,  
      "currency": "usd",  
      "default_source": "ba_1KYy8t2eZvKYlo2CmoRl2hDg",  
      "delinquent": false,  
      "description": "Sample user",  
      "discount": null,  
      "email": "pattie.satterfield@example.com",  
      "invoice_prefix": "44E2E64",  
      "invoice_settings": {  
        "custom_fields": null,  
        "default_payment_method": "pm_1KYy5m2eZvKYlo2CVUMn7KMW",  
        "footer": null  
      },  
      "livemode": false,  
      "metadata": {  
        "order_id": "1234"  
      },  
      "name": "Pattie Satterfield",  
      "source_type": "card",  
      "status": "active",  
      "type": "individual"  
    }  
  ]  
}
```

# PUBLISHING

Documentation Types

## TUTORIALS

LEARNING-ORIENTED

—Most useful when we're studying—

UNDERSTANDING-ORIENTED

## EXPLANATION

Practical steps  
—Theoretical knowledge

“

There is a secret that needs to be understood in order to write good software documentation: there isn't one thing called documentation, there are four.

Divio

## HOW-TO GUIDES

PROBLEM-ORIENTED

—Most useful when we're working—

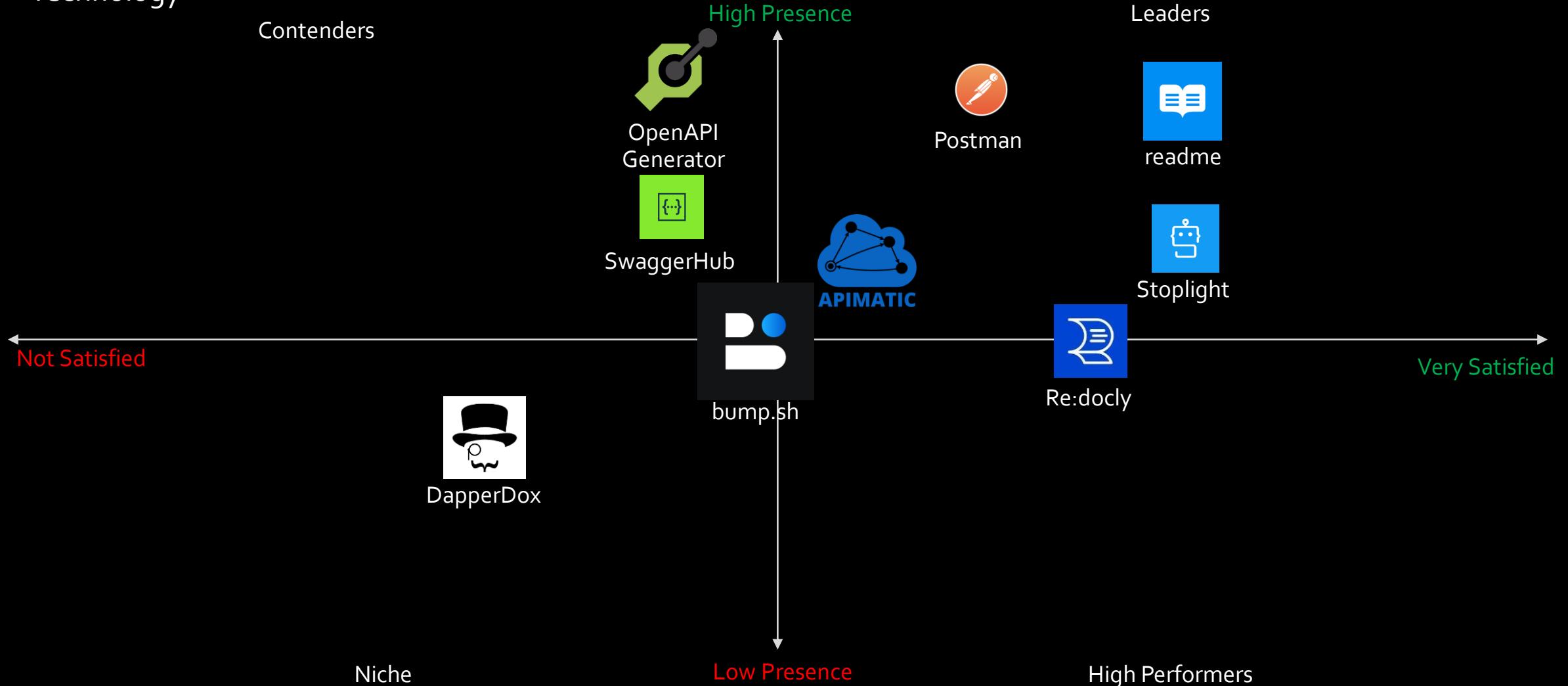
INFORMATION-ORIENTED

## REFERENCE



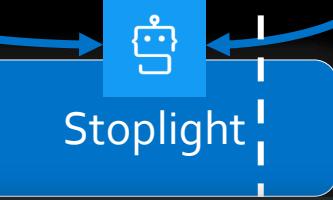
# PUBLISHING

## Technology



# PUBLISHING

SPS API Catalog & Portal

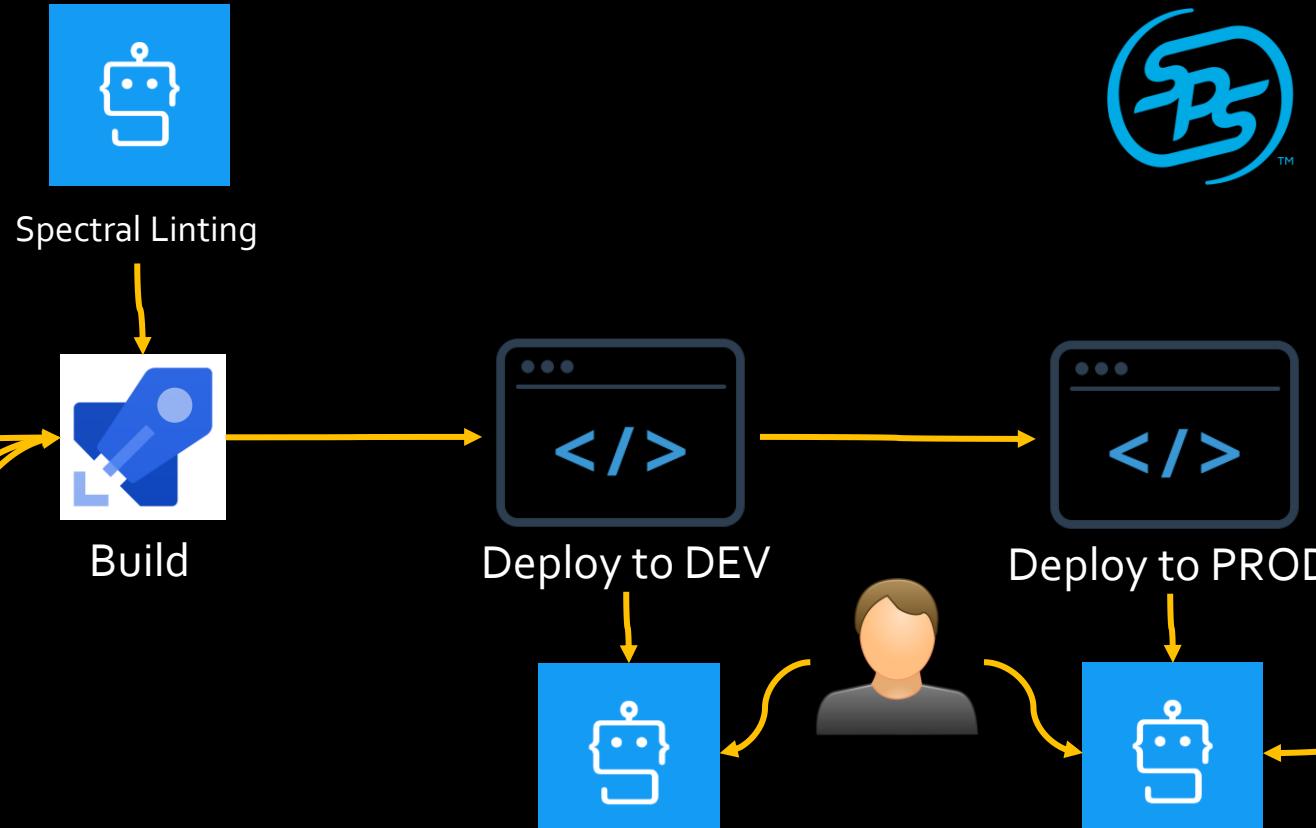


The screenshot displays the SPS Commerce Dev Center Portal interface. At the top, there is a navigation bar with links for Home, Applications, API Resources, Integration Resources, and System Integrations. Below the navigation bar, a banner reads "Building Applications Made Easy". On the left side, there is a sidebar with sections for ANALYTICS (Rule Override Impact), GENERAL USAGE (Authorization, Errors, Collections, Conflicting Updates), and API REFERENCE (OpenAPI Specification, Rule Types, Rules, Rule Instances, History, Analytics, Deprecated, Schemas). The main content area shows a search bar and a "WELCOME TO DEV CENTER" section with links for Get Started, API Schemas, Error Response Schema, Error Message Format, AUTHENTICATION (Getting an Access Token, Refresh Tokens), and REQUESTS (Create a new Instance of a Rule, Get Instances of a Rule, Change the archival status of a Rule, Get a specific Rule Instance, Get active Rule Instances). To the right, there is a "Search history of data manipulation" section with a "Request" panel for a GET /external/v5/history endpoint. The request panel includes fields for after (string<date-time>), limit (integer<int32>), offset (integer<int32>), and until (string<date-time>). Below the request panel, there is a "Send Request" button and a "Mocking" section. At the bottom, there is a "Response Example" section with sample code in C# using RestSharp:

```
var client = new RestClient("https://localhost:8100/external/");
var request = new RestRequest(Method.GET);
request.AddHeader("Content-Type", "application/json");
IRestResponse response = client.Execute(request);
```

# PUBLISHING

Automation



```
# deploy to dev Public 'dev' Version
- template: deploy.stage.v1.yml@templates
parameters:
  environment: dev
documentation:
  markdown: docs/*
  openApi:
    spec: api.oas.yml
    baseUrl: api.test.com/example
  Public 'prod' Version
```



SPS COMMERCE

# RESOURCES

2021 State of the API Report



[postman.com/state-of-api/](https://postman.com/state-of-api/)

<https://stoplight.io/>

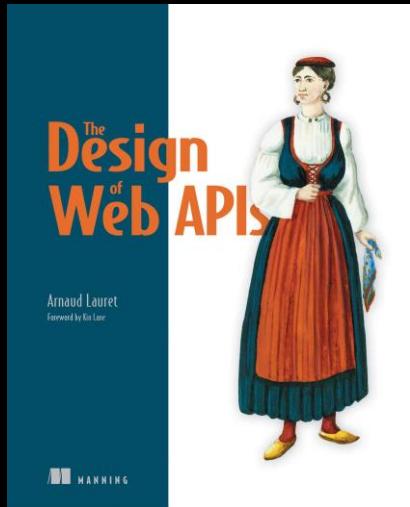


# Stoplight

## API ROADMAP

### API Intersection

Episode 23  
Travis Gosselin  
Principal Software Engineer, SPS Commerce



The State of API Report | 2020



## OpenAPI.Tools

We want to keep API developers up to date with the best OpenAPI tooling around, and help direct folks to high quality modern tooling, instead of being stuck on old v2-based rubbish.

[Contribute](#) [APIs You Won't Hate](#)



## SPS COMMERCE

[smartbear.com/resources/ebooks/the-state-of-api-2020-report/](https://smartbear.com/resources/ebooks/the-state-of-api-2020-report/)

[github.com/SPSCommerce/sps-api-standards](https://github.com/SPSCommerce/sps-api-standards)

“ APIs Are Forever ”



### APIs Are Forever

Once an API is created, it should never be deleted, or changed. “Once you put an API out there, businesses will build on top of it,” Vogels said, adding that changing the API will basically break their businesses.

Werner Vogel’s – 6 Best Practices for API Design

### IS YOUR API READY FOR FOREVER?

Alignment

Enablement

Collaboration

Guidance

ARE YOU SUCCEEDING IN API DESIGN AT  
SCALE?

# API DESIGN FIRST SUCCEEDING WITH API GOVERNANCE



SURVEY



TRAVIS GOSSELIN

[travisgosselin.com](http://travisgosselin.com)



[linkedin.com/in/travisgosselin](http://linkedin.com/in/travisgosselin)



@travisjgosselin

