A journey of building enterprise grade, high transaction rate, resilient APIs

API World, San Jose 2019

Shobha Ramu Shivaiah
Development Manager, UPS APIs

©Copyright 2019 United Parcel Service of America, Inc. All Rights Reserved

Agenda

Characteristics of large-scale enterprise APIs

Evolution

Backward Compatibility

Resilience

Devops applied to API Lifecycle Management

©Copyright 2019 United Parcel Service of America, Inc. All Rights Reserved

Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

Characteristics of Enterprise APIs

Support for large-scale usage

Backward compatibility

Support diverse & geo-distributed customer-base

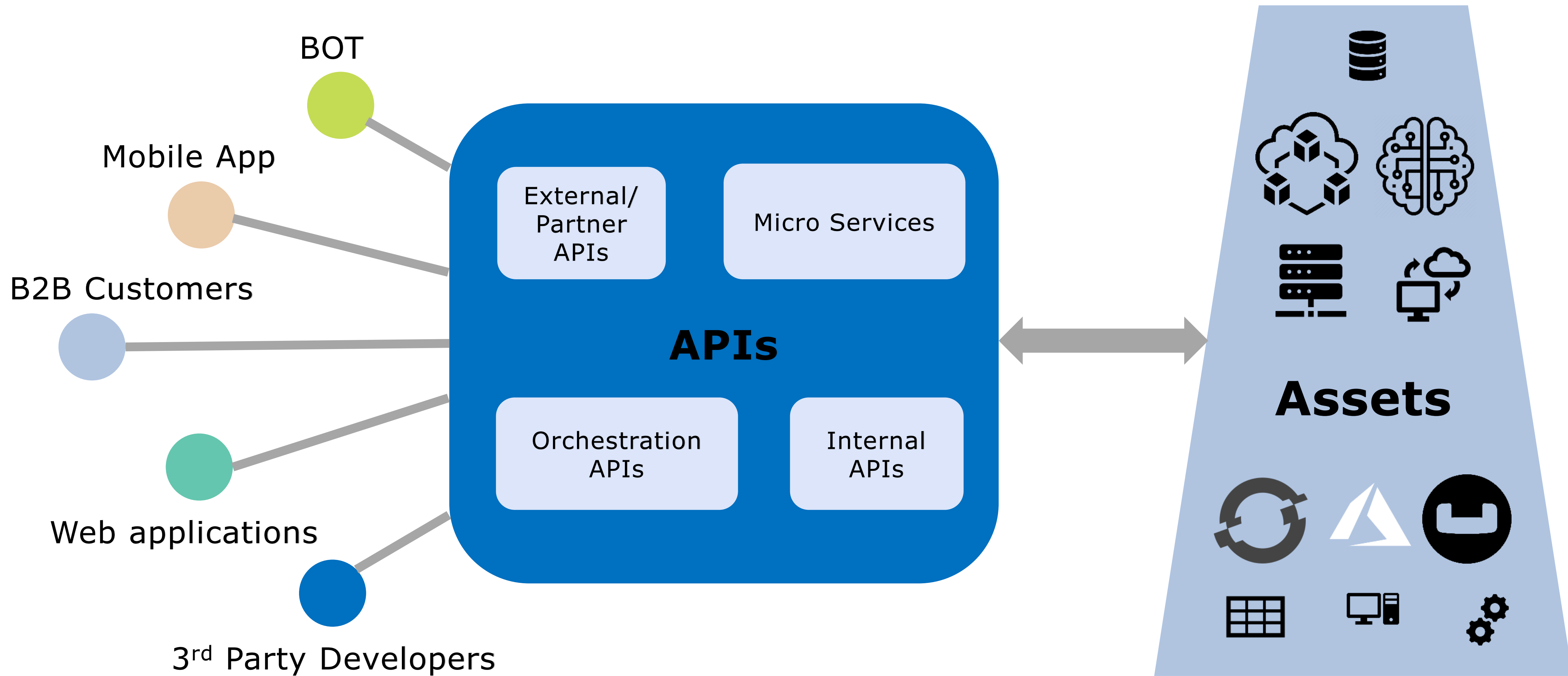
Higher expectations around uptime & need for faster time to market

Secure, compliant and respects privacy regulations

Support existing customers & continue to evolve

Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

Large Scale Enterprise APIs



©Copyright 2019 United Parcel Service of America, Inc. All Rights Reserved

Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

Evolution of APIs

Late 90s — XML over HTTPs

Early 2000s — SOAP over HTTPS

2010 + — REST APIS

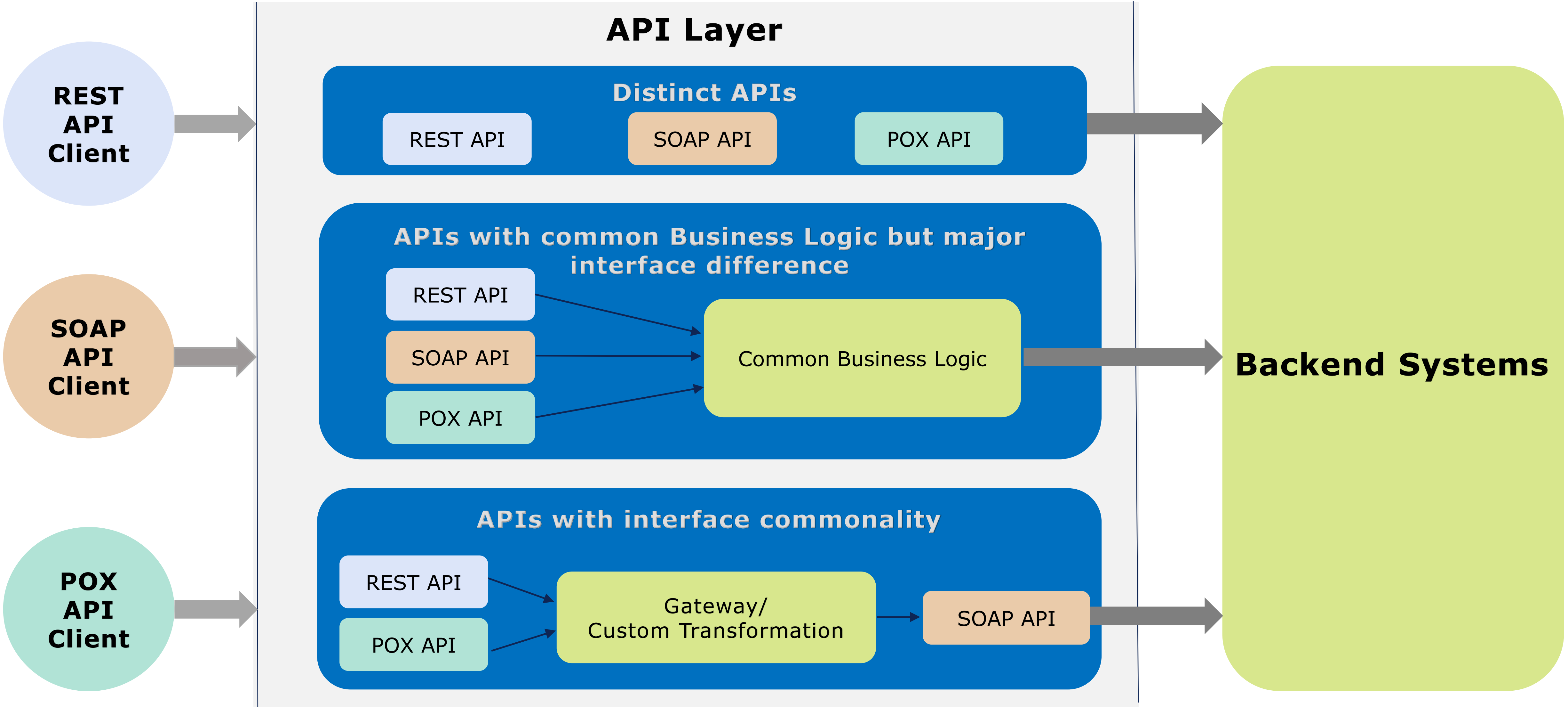
	XML over HTTPs	SOAP over HTTPS	REST APIS
Main Features	• **Plain Old XML (POX)** • **Schema / DTD**	• **SOAP/UDDI** • **Service /function driven** • **WSDL**	• **HTML, JSON, XML** • **Resource / Data driven** • **Optional WADL**
Pros and Cons	• Simple architecture • Less backward compatibility issues • No WSDL dependent client code	• Heavy-duty approach • Built in retry/security mechanism • Advantage of having message meta data	• Lightweight approach • Flexible • Better performance • Simple and easy

Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

API Evolution

1. Stay ahead in technology
2. Be customer obsessed
   a. Don't break existing customers
   b. Don't force unrealistic migration on customers
   c. Account for all customer types
3. Evolution does not mean extinction of existing types
   a. REST is a popular choice for APIs but legacy support for SOAP may be vital to run your business

Sample Scenario - Supporting Multiple API styles



Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

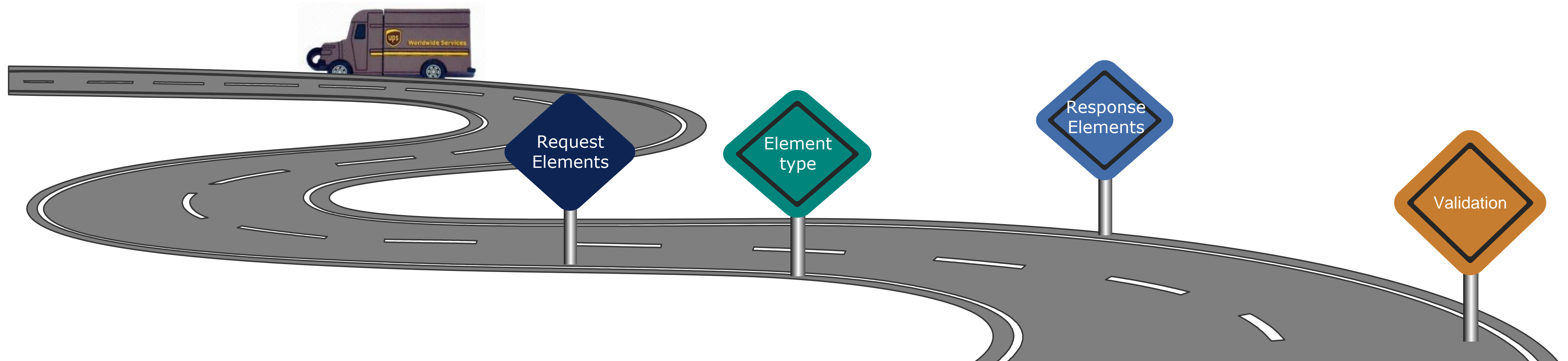
Backward Compatibility - Driving the API Truck

Look Forward :

- Look forward through the windshield for what's ahead
  - Add new features, Support Rest, Graph QL, MicroServices

Look Back :

- Equally important to look back in the rear window and side mirrors
  - Continue to support existing protocols
  - Add features to existing APIs without breaking your customer



Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

Backward Compatibility Principles

Happy Provider

- It is ok to support a **single version API** and not break it
- When supporting multiple versions, offer new features only in new version as an **incentive** for consumers to move
- When you version, make sure you consider **maintenance cost** to keep up with multiple versions
- **Don't be surprised**. Know your customers well
  - Log their payload (*wiping the sensitive data*)
  - Log their http headers
  - Use rich production sample to build your canary tests to put a release with no impact to consumers

Happy Consumer

- Be backward compatible and **avoid versioning** unless it is absolutely needed
- When you retire old versions, be **graceful** and **gradual** on your consumer
- Follow excellent communication mechanism for unavoidable, **mandatory changes** such as *PCI, Tax and regulatory changes per government, TLS and Security updates*

Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

A dive into Backward Compatibility with Examples

- Changing the format of a field – filename *MMDDYYYYHHmm to MMDDYYYYHHmmss*

- Infrastructure changes around the APIs
  - Default encoding in response content type
    - ➢ Weblogic : Content-Type –> text/xml
    - ➢ Tomcat : Content-Type –> text/xml;charset=ISO-8859-1

  - Gateway layer in front of your API
    - ➢ Check if API gateway is being restrictive on the http header contents like MIME TYPEs.
    - ➢ Test with all possible HTTP request headers to compare before and after
    - ➢ Compare the before and after HTTP Response headers

- Changing the error code for a given condition - Clients could have specific logic based on the error codes

Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

Versioning

1

Versioning a Rest API
- Through URI Path, query parameters, Custom headers, content negotiation

2

WSDL Namespace versioning, Versioning using an element in the XML or JSON body

3

Soft versioning:
- Pair a response change with a request change when applicable

4

Documentation
- Share information about your versioning strategy to stop clients from building fragile integration
- Remove deprecated elements and versions from documentation

5

Visualize the future changes when designing your API.
- Be as restrictive as needed in the initial release
- Use code values where needed to accommodate future new types

Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

Resilience – Monitoring and API implementation

Monitoring

Enrich your API code to include monitoring around backend calls

Example : Java APIs JMX : Pull model; Netcool : Push Model;

- Have monitoring tools that can dissect backend calls

Trend your error codes not just HTTP status

Outside In monitoring – Have good optics into your APIs from external perspective

API Implementation

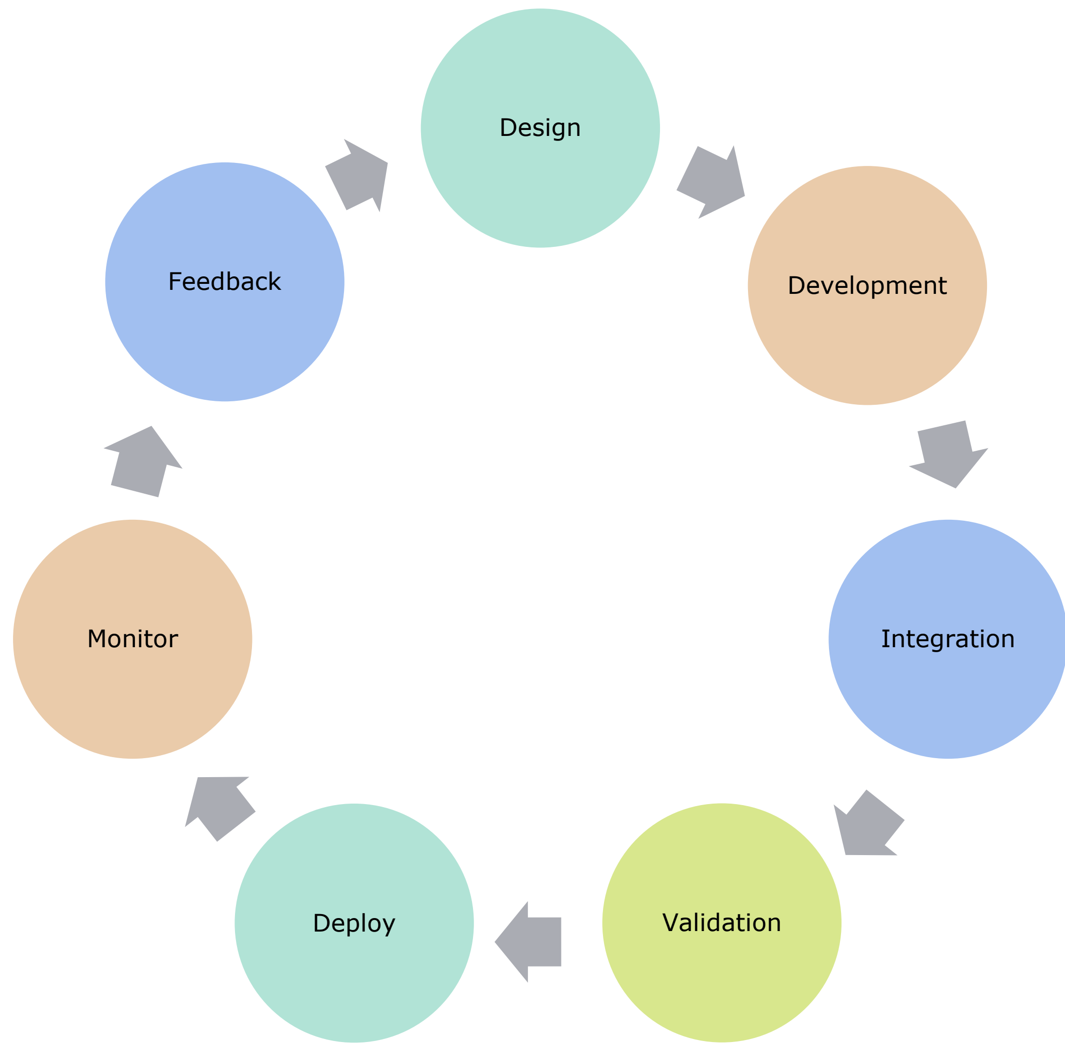
Not all backend failures need to cause your API to fail – Partial response

Blind retry logic can add more stress to your APIs during issues

Rich logging to pinpoint the bottle necks in your APIs during issues

Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

DevOps applied to API Lifecycle



Design & Development

Instant visibility for parallel changes – Parallel API test plan from schema, JSON, swagger specs

Design and code for shorter & iterative API changes

Automate build to run a set of golden

All artifacts and dependencies are automated in version control – Bean objects, YAML, Swagger files

consumers (internal) where possible

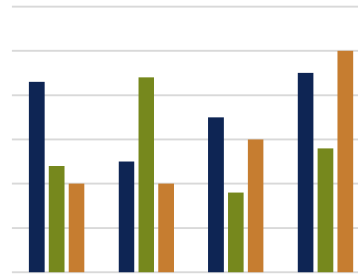
Any commit to version control triggers the automated recreation of APIs.

Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

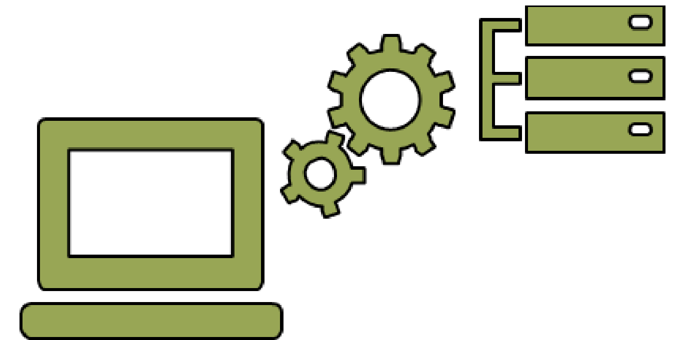
Successful API teams

Understand your API Consumers

KNOW your API Metrics



Continuous Innovation



Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.

APIs are here to stay and evolve.

API is a journey not a destination.

Thank You

(LinkedIn : Shobha Ramu)

©Copyright 2019 United Parcel Service of America, Inc. All Rights Reserved

Proprietary and Confidential: This presentation may not be used or disclosed to other than employees or customers, unless expressly authorized by UPS.
© 2019 United Parcel Service of America, Inc. UPS, the UPS brand mark, the color brown are trademarks of United Parcel Service of America, Inc. All rights reserved.