API Management (APIM)

Strategy & Governance Offering

Overview

NA Strategy Services Team – Arnold Abernathy , John Cocke

March, 2018

john.cocke@ca.com

ca technologies

Introduction

CA APIM Strategy Services bridges the gap for Customers that may have already benefited from an API Academy, early proof of concept projects, and are now in need of an appropriate APIM governance framework, to help you make the leap to development and implementation.

We provide a 2-3 day Workshop or an extended 3 – 4 week engagement, often working in a team with partners and CA Professional Services.

Our scope is enterprise-wide, however we focus on customer use of the CA APIM suite, APIM Strategy (vs. API Strategy), long-term architecture vision and lessons learned from prior implementations.

We shape each workshop or engagement based on canvassing each customer’s As-Is/To-Be business and technical goals.

Agenda

Topic	
Introduction	
Our Understanding	
API 360 Overview	
Alignment & Usefulness ***<u>Agenda</u>*** - **API Life Cycle** - **APIM Program** - **APIM Governance (Charter)** - **APIM dependencies and impact**	*<u>Related topics</u>* - *Formalize an API Strategy* - *Inventory of target API audience* - *Define desired-state based on business objectives* - *Impact analysis of API dependencies (SOA & Legacy)* - *Setup Program Governance*
Engagement & Usability ***<u>Agenda</u>*** - **Developer Program** - **Developer Experience (DX)** - **Consumer toolkits** - **Provider playbooks (e.g. CD & CI)**	*<u>Related Topics</u>* - *Identify Developer communities (external & internal)* - *Establish Developer Evangelist role* - *Create/rollout Developer Framework, including:* • *Case studies outlining integration types* • *Developer community communication plan* • *API Portal* *Note: Developer Evangelist role owns Developer Framework*

ca technologies

Agenda

Topic	
Scalability & Evolvability ***Agenda*** - **API Profiling** - **Continuous integration, Deployment** - **DevOps culture** - **Leveraging APIs and Microservices** - **Microservices Design and Readiness**	Related Topics - *Rationalize tactical and planned APIs for predictive reliability,* - *Consider the following design options:* • *"fast lane / slow lane"* • *Internal vs. external Gateways* • *Gateways local to certain APIs and services geographically* • *Runtime service discovery*
Manageability & Sustainability ***Agenda*** - **API/ESB Interlock** - **Developer-tooled Monitoring** - **Incident Management** - **Demand Management** - **Configuration Management** - **Service Catalog Management** - **Capacity Management** - **Information Security Management**	Related Topics - *Provide optics into API technical processing specific for each stakeholder group* - *Provide optics into API business processing specific for each stakeholder group* - *Align CI-CD approach with existing SDLC* - *Awareness and buy-in for SLAs enforcement with tactical and planned API owners (e.g. timeouts, concurrency/throttling, etc.)?* - *Rollout ability to do static and dynamic security testing as part of an API's SDLC*

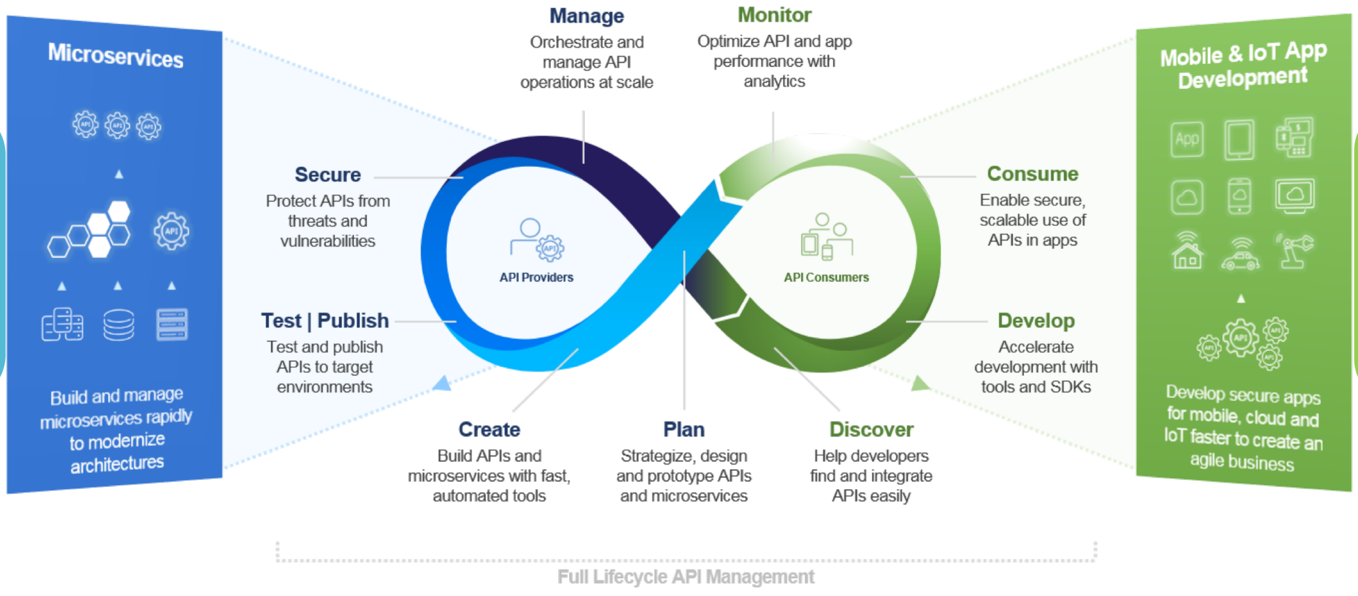
Alignment & Usefulness

API360 Overview

Formalize an API Management Strategy

APIM Monitoring / Operational Maturity
(Manageability & Sustainability)

APIM Velocity of Change / CI / CD
(Scalability & Evolvability)



Microservices
Build and manage microservices rapidly to modernize architectures
Manage
Orchestrate and manage API operations at scale
Monitor
Optimize API and app performance with analytics
Secure
Protect APIs from threats and vulnerabilities
Test | Publish
Test and publish APIs to target environments
API Providers
API Consumers
Consume
Enable secure, scalable use of APIs in apps
Develop
Accelerate development with tools and SDKs
Create
Build APIs and microservices with fast, automated tools
Plan
Strategize, design and prototype APIs and microservices
Discover
Help developers find and integrate APIs easily
Mobile & IoT App Development
App
Develop secure apps for mobile, cloud and IoT faster to create an agile business
Full Lifecycle API Management

APIM Developer Framework / Evangelism
(Engagement & Usability)

APIM Strategy / Vision / Governance
(Alignment & Usefulness)

ca technologies

Alignment & Usefulness

API Program

Build an API Management Framework

API Program Charter (Deliverable)

- Purpose
- User Stories
- Roles
- Participants
- Adoption
- API Branding Approach (LOB vs Enterprise)
- Business Canvas (roadmap to business objectives)
- Workstreams and Deliverables (API Management roll-out)
- KPIs for APIs (Business, Traffic, Developers, Service, Marketing, Support)
- Inventory and Validate API audience including direct feedback
- API Management Key Functions
- Program Charter History (Revision Control)

Alignment & Usefulness

API Strategy Project Plan

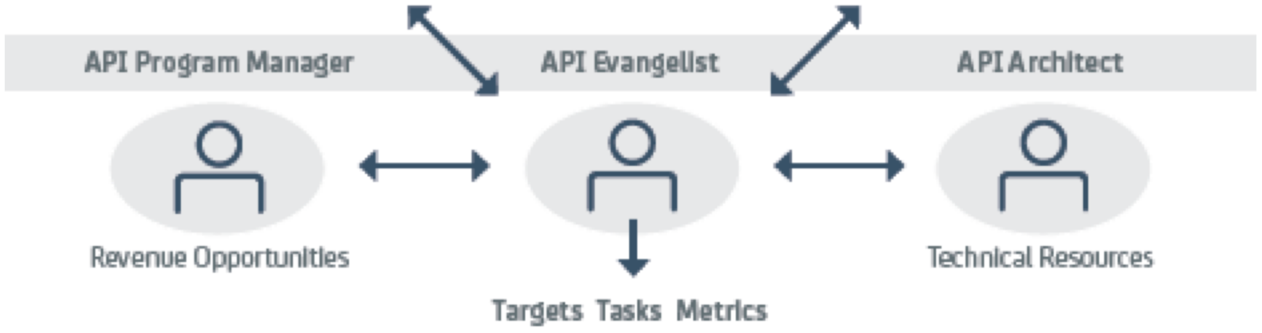
	Alignment & Usefulness	*Engagement & Usability*	*Manageability & Sustainability*	*Scalability & Evolvability*
	API Business Survey	API Current Capabilities Architecture	Future API Capabilities Architecture	**Establish API Program**
Activities	• Working sessions with the following stakeholders to discover business objectives, potential challenges, and perspectives on governance: • Review of key projects and initiatives associated with Mobility • Identify inter-dependencies • Estimate complexity and level of effort for each initiative and project	• Working sessions with key shared IT and BU groups to discover current API capabilities and challenges • Working sessions with key developer groups to discover current API capabilities and challenges • Mobility readiness assessment • Discover in-flight IT initiatives and timelines	• Collaboration with agreed upon IT stakeholders regarding future-state technology architecture	• Assistance with API Program Charter, including: - Vision - Roles and responsibilities - High-level process definition - Communication plan • Assistance with API standards and guidelines
Outputs	• *API business survey* • *API Roadmap and High-level Plan*	• Current-state API architecture	• Future-state API architecture	• API Program Charter • Strategy Roadmap

ca technologies

Engagement & Usability

Developer Evangelist

The Developer (API) Evangelist



In addition to creating stellar APIs, the API evangelist is the person raving about it to their colleagues, tweeting about interesting stuff it can do, or maybe even talking about it with customers directly, as appropriate.

Below are some channels being used by the API evangelist to promote your APIs:

Internal	Partner	External
On the dev portal	Signup emails	Signup emails
Team meetings	Dev Twitter accounts	Twitter
Employee guides	Meetings with developers from other companies	StackOverflow
In the hallway	SDK workshops	GitHub
Slack channel	Revenue use scenarios	Conferences

Engagement & Usability

Developer Experience

Objective: Provide an environment, information, and tools that assures that developers will adopt your APIs

Considerations

External (consuming)

- How do developers get started?
- How do developers learn how any of our APIs work?
- How do developers troubleshoot problems and engage with our internal community?

Internal (providing)

- How do we ensure that our systems are safe?
- How do we restrict usage?
- How do we see how our APIs are being used?

Industry Perspectives

- Developer built monitoring is preferable to configuration or architecture boards
- Developer Portals enhance velocity and API management
- Strong communications with developer communities are essential
- Providing SDKs enhance API usage, (i.e mobile security)
- Document APIs with an API definition language like Swagger or OpenAPI, using JSON or YAML depending on whether you will write or write/generate
- Make testing APIs easy using Continuous (automated)Testing
- Provide high-value monetized APIs, this will validate their usefulness to the provider

Functions, Processes, Technology

"As Is"

- API strategy or roadmap?
- Current State standards:
  - Communication protocols in use (i.e. REST vs custom)
  - Access point characteristics
  - Uniform interface
  - Transport language (i.e. JSON)
  - API extensibility

"To Be"

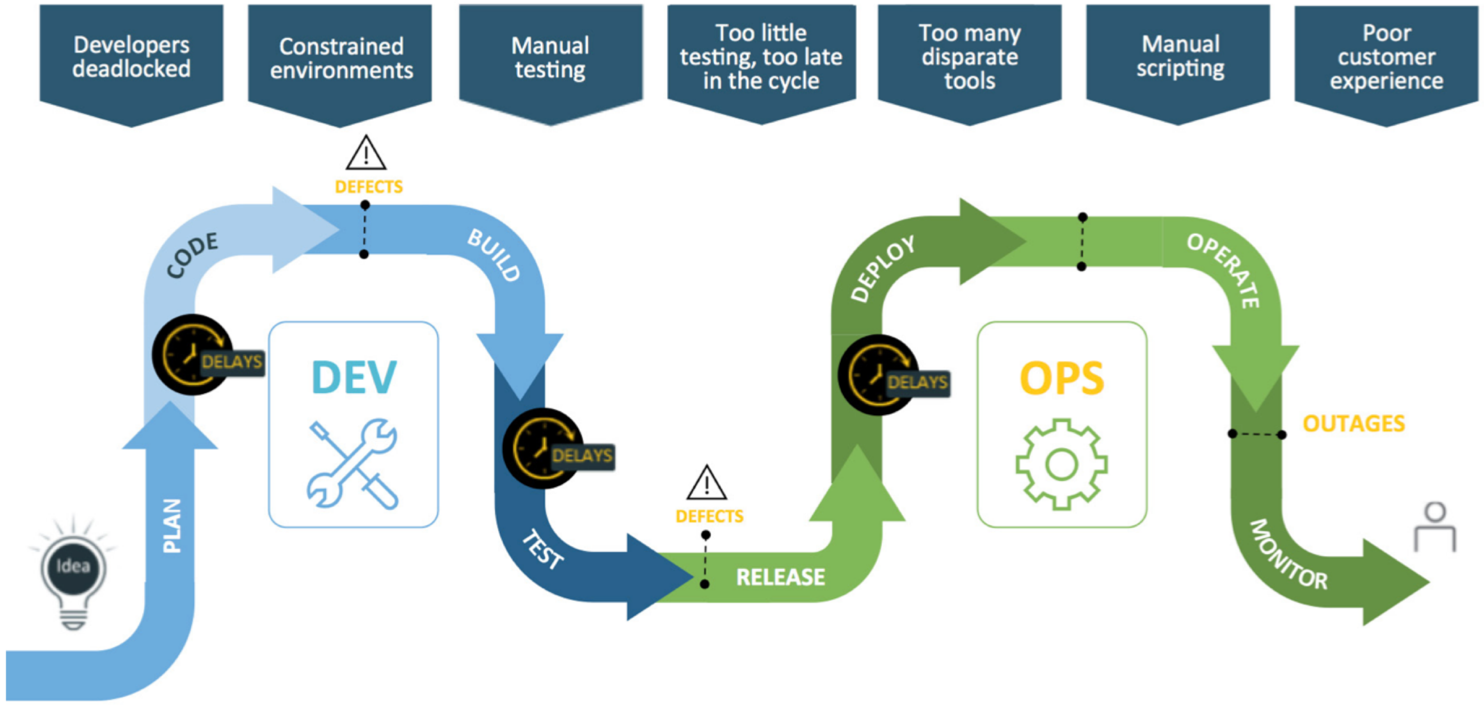
- Use developer portal to centrally expose developer-facing, design time artifacts (such as WSDLs/WADLs, schemas, metadata, business rules, etc)
- Centrally manage API usage
- Centralize internal API documentation with a synchronized CMS and Service Catalog, using existing CMDB if there is on

Scalability & Evolvability

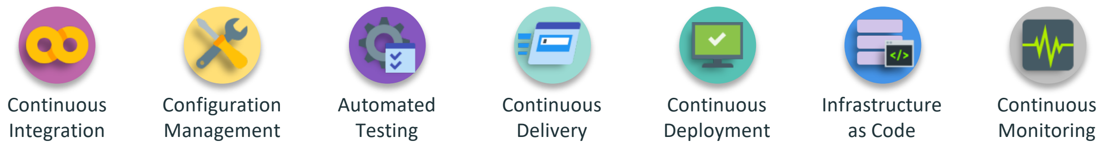
DevOps

What are we solving with **DevOps __culture__** ***in your world***?

Separation of development and operations (people, process and technology) slows application delivery and impacts quality.



Key DevOps Practices ===.>



ca technologies

Manageability & Sustainability

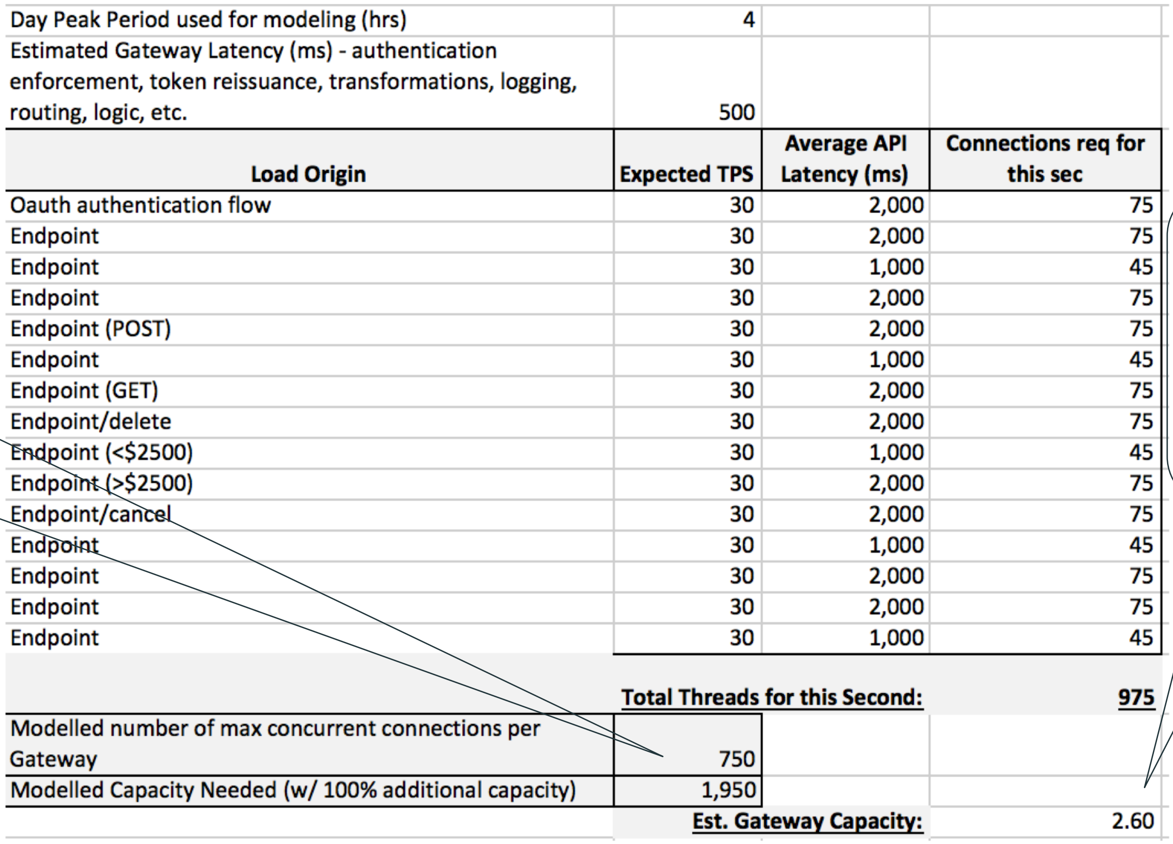
Monitoring API Performance

Developer Tooled Monitoring

- The developer knows HOW the API is supposed to achieve its objectives
- Developer created tests:
  - **Performance-oriented:** These are tests that make individual calls to each and every method your API provides. If a response takes longer than a specified time limit, it means that there's a problem on that specific endpoint.
  - **Functional testing:** This type of testing works by making singular calls to each API method to thoroughly test every API function, using different kinds of payloads, or even sending data that will produce errors. Responses are then compared to the expected behavior to locate errors.
  - **Use case testing:** This is a more sophisticated type of test and can be achieved by combining calls to different endpoints into a single test. Each test should expect a specific response and execution time limit.
  - (beyond the APIM Policy Manager)

Manageability & Sustainability

Capacity Management



Day Peak Period used for modeling (hrs)	4		
Estimated Gateway Latency (ms) - authentication enforcement, token reissuance, transformations, logging, routing, logic, etc.	500		
Load Origin	**Expected TPS**	**Average API Latency (ms)**	**Connections req for this sec**
Oauth authentication flow	30	2,000	75
Endpoint	30	2,000	75
Endpoint	30	1,000	45
Endpoint	30	2,000	75
Endpoint (POST)	30	2,000	75
Endpoint	30	1,000	45
Endpoint (GET)	30	2,000	75
Endpoint/delete	30	2,000	75
Endpoint (<$2500)	30	1,000	45
Endpoint (>$2500)	30	2,000	75
Endpoint/cancel	30	2,000	75
Endpoint	30	1,000	45
Endpoint	30	2,000	75
Endpoint	30	2,000	75
Endpoint	30	1,000	45
	Total Threads for this Second:		**975**
Modelled number of max concurrent connections per Gateway	750		
Modelled Capacity Needed (w/ 100% additional capacity)	1,950		
	Est. Gateway Capacity:		2.60

Configurable per Gateway & dependent upon available Gateway resources (e.g. CPU, RAM, etc.)

Approximate number of Gateways needed