Best Practices in Building External API Programs

Rapid

APIs as products

Whether their goals are to accelerate innovation, increase revenue, or extend product lifecycles, today's digital transformation initiatives rely on APIs. But for an organization looking to become a leading digital enterprise, simply having APIs is not nearly enough. To fully execute on their digital transformation goals and objectives, leading enterprises need a clear API strategy built on a productized approach to external APIs.

The following details the key steps and best practices for productizing APIs. These recommendations provide guidance on defining APIs as products, building the framework to operationalize your API program, and designing and metricing a GTM plan.

Objectives for Productizing your APIs

Before diving into how to productize your APIs, it's critical that business and technical leaders establish and agree upon the objectives. This shared understanding and cross-functional agreement will help inform value messaging to external audiences, the communication plan, and best practices around the operational aspects of the program.

For external programs the most common objectives include:

- **Accelerating innovation**
- **Decreasing time to market**
- **Building new revenue stream and external product lifecycles**

These objectives (among others that your organization may determine) should serve as the core criteria for defining and recognizing successful execution of your API program. With these objectives in mind, project and organizational leaders can begin outlining and planning for how they will contribute to and benefit from this collective approach to an API program which views, operationalizes, and leverages APIs as products.

4 Steps To Take To Define APIs as Products

For many organizations, especially those that have not been software-first companies historically, APIs have been viewed purely as a technical tool. Ultimately this is a narrow view of APIs and their utility. To be successful in their transformation into becoming a powerful digital enterprise, organizations need to instill a product mindset when it comes to their APIs - especially those that are intended for external use.

1

Instill a Product Mindset

The first step in building this mindset is to think of developers both as builders (provider in a traditional vendor relationship) and consumers (buyers). Instilling this mindset forces those APIs builders to clearly define the function and value of their APIs. The clarity derived from these definitions will not only make it easier for consuming developers to understand how they can benefit from using a particular API but will also help inform some of the operational aspects for that APIs search and discovery. A clear articulation of the API's value and function enables that builder to contribute to more informative documentation. It also ensures that that builder can properly describe and tag that API when it is published to an API hub. This makes it easier for consumers to discover, improving the chances of that API's adoption.

2

Define Requirements

Next, generate a requirements document. This document should be the source of truth for the objectives of the API program and outline the needs in achie ving those outcomes, detailing required tools, budget, and people. This document should outline the function of the program's API and align their value with its top level objectives. Its scope should not be limited to the APIs that are already built, but should also include a view of the net new APIs that will need to be added (along with associated headcount) to achieve the program's goals.

3

Begin in Design

Successful API adoption starts at the earliest stages in the development process with API design. Starting with clear API design principles ensures an intuitive and consistent experience that helps ensure API adoption.

4

Solicit Developer Feedback

Instilling a product mindset with external APIs requires focus on collecting and addressing developer feedback. Prioritizing developer feedback has two key benefits.

- **Identify Bugs and Unexpected Behaviors**
- **Identify and Prioritize New APIs**

With a firm grasp on how to think about and build APIs as products it's time to start looking at how to effectively operationalize their use across the organization with proper resourcing of both personnel and tooling.

Resourcing People and Tools

The responsibility to productize APIs is not solely reliant on the developers who are building them but rather the entire organization to enable the API program with the tools, resources, and structure to realize its potential and meet objectives. Critically important is the organizational structure that will oversee the API program.

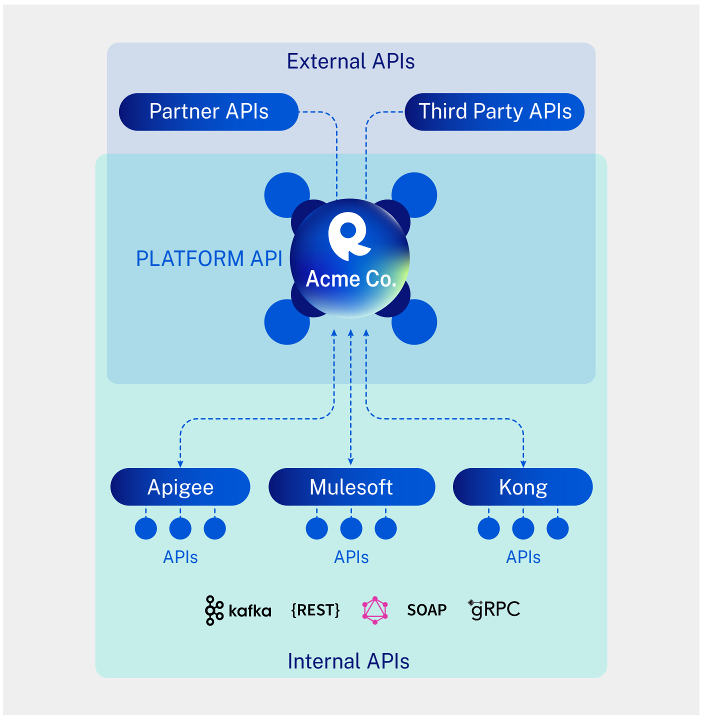
Organizational Structure

There are three common structures which an organization can elect to deploy: single team ownership, horizontal crossorganizational, or a hybrid structure (which combines some aspects of the two).

	Pros	Cons
Single Team Ownership Accountability and Autonomy	• Greatest level of autonomy • Creates a sense of ownership • Allows for a greater degree of experimentation	• Can hinder participation from the broader organization
Cross-Opperational Investment and Support	• Streamlines interoperability • Leverages specialized expertise • Fosters investment in program success	• Reduces overall agility • Can obscure individual roles and responsibilities
Hybrid Model Best of Both Worlds	• Most ßexible overall structure • Can take advantage of both efÞciencies in descision making and buying power • Allows for both autonomy and collaboration	• Requires clearest boundaries in decision making and responsibility • Can create bottlenecks in descision making

Selecting An API Hub

Having an effective API hub that enables all of its users is absolutely essential to ensuring that the goals for the API program are met. This hub is the place where all of your developers and external API consumers should be able to come and access all of the APIs and tools they need to discover and use all of the APIs available to them



Requirements for an API Hub:

Centralizes All APIs – Creating a centralized place where developers can discover, collaborate on, and use all of the available APIs is perhaps the most critical element of what an API hub has to deliver. Without this capability developers are forced to jump from place to place to find all of the APIs available to them, which slows the development process, hinders adoption, and can ultimately lead to the failure of API programs.

Most enterprises have a number of API gateways comprising their API ecosystem. To be effective, an API hub must be able to work with this multi-gateway architecture to centralize all of the APIs into a single place. Having this centralized location simplifies the discovery process for developers and helps ensure API adoption.

Optimizes Developer Experience – But that API discovery is only part of the developer experience. A good API hub needs to support the entirety of the developer experience through integrated developer tools for API design and development, to testing and publishing. Having this breadth of capabilities integrated into the API hub allows for the developer to stay within the same context. Additionally this intuitive experience should be designed to facilitate collaboration amongst developers across the entirety of the development lifecycle and support their use of multiple API types including REST, SOAP, GraphQL, and Kafka.

Ensures Governance and Access Controls – There are two primary things to look for when assessing an API hub's requirements in governance and access controls. First, is its ability to provide a simple and intuitive onboarding experience for those developers subscribing to APIs, one of the most critical elements in ensuring API adoption. Second (but equally important) is its ability to ensure governance and access controls across the entirety of the API ecosystem. To be effective, an API hub must be able to establish visibility and role-based access controls across all APIs, regardless of the gateways which they might be running through, and provide clear insights into who is using them and at what level.

Offers Seamless Integrations – When looking at integration capabilities the most important thing to evaluate is an API hub's ability to provide developers and architecture teams with the flexibility to choose the best technology for their need or use case. Delivering this requires an open-platform built on the principles of enabling developer experience. It should therefore offer extensibility through a platform API, support for webhooks as well as CI/CD integrations, and again be constructed to support multi-gateway API ecosystems.

Building a GTM Plan

Simply making APIs available does not ensure their adoption. A productized approach to APIs requires a well designed GTM plan. This plan should combine tooling for everything developers need with a deliberate communication plan which articulates the utility and value of all the APIs. Importantly this plan should also provide guidance and instruction on how to most effectively engage with those APIs. There are a couple of key steps in designing and delivering this.

1

Align on an API Hub

Until the organization has a single source of truth for all of its APIs, the developer experience will continue to be fragmented. For guidance on how to evaluate an API hub's ability to deliver this see the above "Selecting An API Hub" section.

2

Select An Initial API Set

The next step is to decide on which APIs will be included in the initial o ffering, those which will be added in subsequent releases and prospective timelines of their delivery. Starting with a subset of the APIs that will ultimately become available limits the number of variables in assessing the program's performance. With a discrete set of APIs it is easier to see trends in varying levels of adoption and usage, isolate reasons for that disparity and make changes both to that initial set as well as those subsequent ones which will be added.

3

Outline User Journeys

Lastly, it's important to outline the complete user journey. This applies to both API builders and API consumers. Having a clear view of both of these audiences will enable a more effective communication plan that speaks to what is important to them and offer clear guidance on how they can interact with APIs in the API hub..

Stages and Objectives of a Release Plan

Like any product, the APIs that are offered to external parties such as partners should grow through set stages of product development. Each with their own set of definitions, goals, and criteria for success.

	Early Access	Preview	Beta	GA
Audience	Internal Developers	Trusted Partners	Early Adopters (Private Invite or Public)	Target Market
Goal	Ensure Proper Functionality	Test Scale	Validate Markets	Widespread adoption
Objectives	• Validate established API functions • Identify bugs and unexpected behaviors • Identify specific functionality gaps	• Test the API's performance at increased scale • Identify bugs and unexpected behaviors • Solidify specific functionality requirements that will make the API atractive to external users	• Test market assumptions • Validate workflows and effectiveness of the API documentation • Ensure API performance at increased, market level scale	• Acquire a user base • Scale the user base over time • Generate revenue

Establishing A Communication Plan

A well-constructed communication plan should mirror the steps and tenants which were outlined in the GTM plan. This communication plan should clearly outline which APIs are available, what their value is to the user, and how to interact with them.

It's essential that this communication plan takes into account both the API consumers and API builders (each with communication documents handled separately). For API builders, this plan should outline best pr actices and methodology for how to communicate their API's functionality and benefits. Creating this structure at the earliest stages of a communication plan ensures that all participants will operate with a shared nomenclature that is clear and informative.

The communication plan should:

- Provide guidance on how to communicate an API's value.
- Offer instructions on how users can navigate and interact with the API hub.
- Outline processes for API discovery, requesting access, and contacting support.

Evaluating The API Program With Metrics

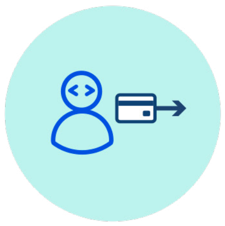
To understand how the API program is performing it is essential to set quantifiable metrics. With those measurements the next step is to set goals identifying what success looks like across them at specific milestones and/or standard intervals such as six months, a year, two years, and beyond. Choose metrics which align specifically with the organization's specific outcomes or goals. Some of the more commonly used metrics include:

- Total API Call Volume – Evaluating the growth and usage of the API over time
- API Call Volume By Partner – Gauging distribution to analyze widespread market adoption
- Average Daily User Count – Understanding the utility and effectiveness of the API
- Total Number of Users – Measuring discovery and adoption
- Total Number of Partners – Indicating level of market adoption
- Revenue – Analyzing effectiveness of monetization model

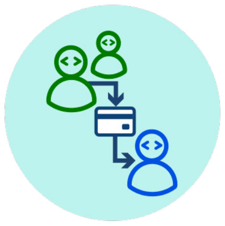
Applying Monetization

When it comes to the API-driven revenue metric, there are a number of models that an organization might pursue as part of its API program. Before deciding on which model to deploy, organizations need to determine the primary objective of the API. Generally these objectives can be assigned one of three categories: data generation, revenue generation, or product stickiness. With data generation, organizations leverage the data derived from third-party applications to inform product or marketing decisions. Revenue generation directly monetizes the API while product stickiness relies on positioning the API as a value add. Oftentimes organizations will apply a mix of these models as some APIs are better suited to one over the other.

Free (Freemium)
No financial transaction or requirement for use



Paid
The user pays some type of fee



Indirect
Revenue is attributed or realized by another product or initiative

Free (Freemium)

In this model, there is no financial transaction between the API providers and its consumers. The primary motive for this type of model is data generation. With data generation being paramount, encouraging adoption amongst developers is critical. The free model helps facilitate this by removing what can sometimes be viewed as a steep barrier of entry. Offering the API for free is thus an effective means to maximize the available dataset. There is however a drawback, because there is no attributable revenue, free APIs can sometimes be viewed internally as a cost driver.

Paid

The most straightforward and common path to monetization is a model in which the API consumer pays. The structure of this model can vary but for organizations whose top priority is API driven revenue, the paid model offers the clearest path to revenue generation and attribution. Companies have the flexibility to still offer some level of free usage, whether that be via a tiered, or pay-as-you-go structure, thus encouraging adoption but will require payment for higher level of usage or functionality. Because these models are charged to developers they require a clear articulation of the value of the API and a high level of product support.

Indirect

Common in partner API programs, the indirect model is perhaps the least str aightforward in its ability to recognize and assign revenue to a particular API. The primary driver for this approach is companies that want to create better stickiness with their products and services. APIs are generally offered as an add-on or included in an application or partner package facilitating integration with partner or customer workflows. These tighter integrations strengthen the relationship by making it more expensive and time consuming to switch from one provider to another.

Conclusion

Successful digital transformation relies on organizations effectively productizing their APIs. Adopting this productized approach helps ensure API adoption, continued use, and drives objectives such as accelerated innovation, faster time to market, and API driven revenue. Leveraging the best practices covered in this guide will help ensure effective API productization and successful digital transformation.

About API Hub Enterprise

Rapid's API Hub Enterprise is the next-generation API hub that enables leading enterprises to accelerate innovation and bring software to market faster with one place to discover, govern, and collaborate on all APIs. Rapid offers a single platform for all APIs – every API protocol, every API type, and across every API gateway. API Hub Enterprise can programmatically communicate with all your API gateways, enabling you to centralize all of your organization's API hub environments to help developers reuse and connect to existing APIs faster while providing IT with enterprise-wide visibility and governance of API consumption.

Rapid

Global HQ
85 2nd Street, Fourth Floor
San Francisco, CA 94105

Contact
info@rapidapi.com
www.rapidapi.com

05/21 - Updated 02/23