

The API Product Mindset

How to move fast, delight customers, and continually innovate to thrive in today's economy



Table of contents

The case for API products	03
How API products provide ongoing leverage	06
The product mindset pillars: outside-in, time to market, and iteration	08
• Take an outside-in approach	08
• Improve time to market with MVPs	11
• Mature products through iteration	13
A guide to product management for APIs	14
• Introducing the API team and the API product manager	14
• Building a minimum viable product: design, develop, secure	17
• Curating a world-class developer experience: publish, evangelize, scale, monitor	20
• Championing the business value: establish KPIs, iterate, monetize	21
What's next?	22

The case for API products

Application programming interfaces, or APIs, are the way software talks to other software—so by definition, they're a technology that connects systems. Many enterprise leaders think of them primarily in these terms. For example, 57 percent of respondents to Google's enterprise digital maturity assessment tool, [Apigee Compass](#), characterize APIs as systems integration technology.

But APIs are much more. They're interfaces that enable developers to repeatedly leverage data, functions, and applications to build new products and services. They're how a business expresses itself via software, and they enable that business to rapidly expand into new contexts or adapt to meet changing user needs and preferences.



Take ridesharing apps. They exist partly because they can leverage existing capabilities made available via others' APIs. When people order a car, a ridesharing app leverages services such as Google Maps APIs for navigation. In turn, those ridesharing companies express their businesses as APIs. This lets developers build ridesharing into new experiences, such as allowing an end user to order a ride via a voice assistant. When the user pays for a ride with a given digital payments platform, an API enables that transaction too.

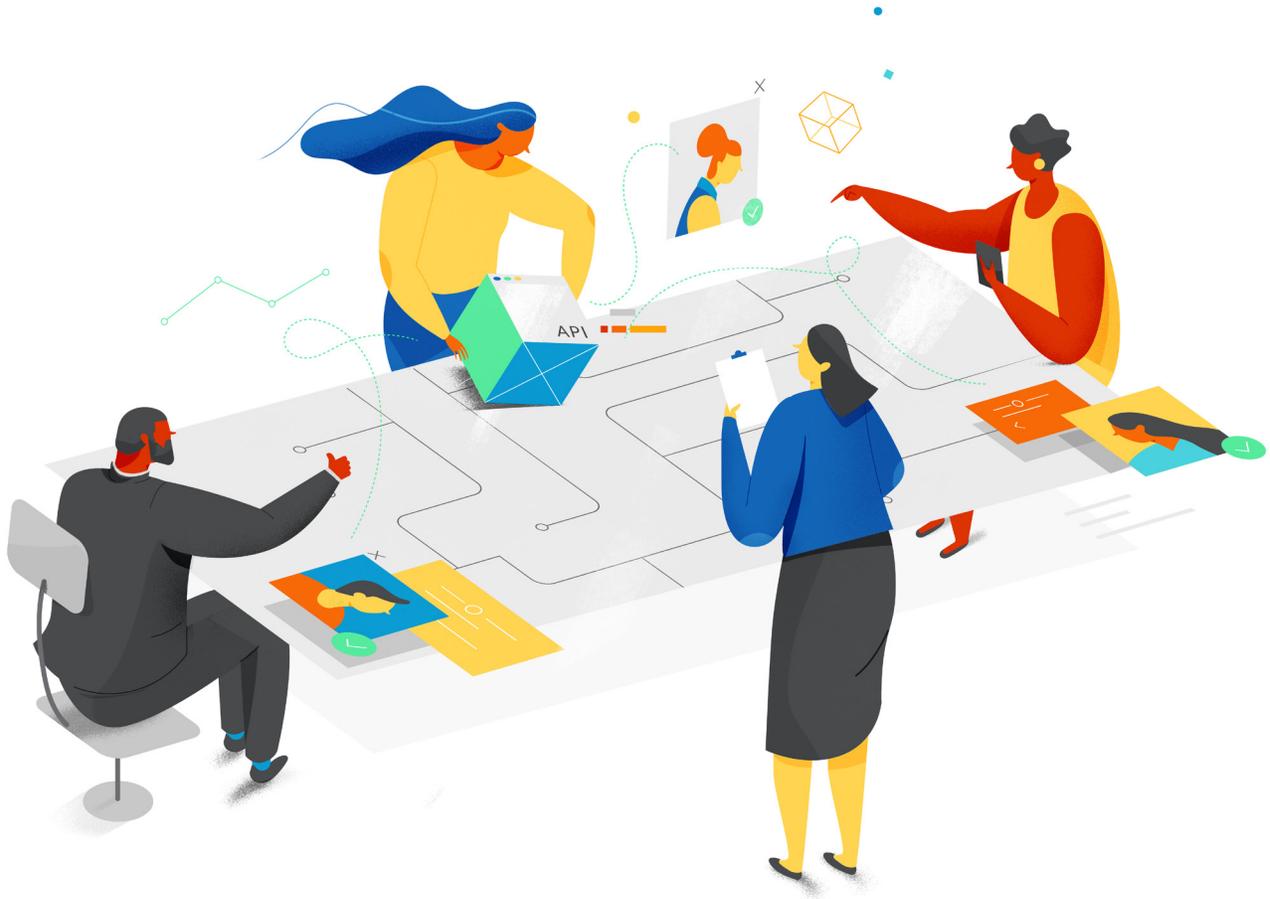
APIs aren't just some back office technical detail, in other words—they're both products for the developers who build today's customer experiences and the mechanisms through which value is increasingly exchanged in modern economies.

In recent years, Salesforce has generated half its revenue through APIs¹. For eBay, the figure has been 60 percent—and for Expedia.com, an astonishing 90 percent². Google customer Magazine Luiza leveraged APIs to replace an old e-commerce model that supported only 35,000 SKUs with a global online marketplace featuring more than 1.5 million. The number of public APIs has grown

¹ Harvard Business Review, The Strategic Value of APIs, January 2015. <https://hbr.org/2015/01/the-strategic-value-of-apis>

² Ibid.

In the Apigee team’s experience working with hundreds of top enterprises, the most impactful API programs are typically led by a team that builds APIs as products designed to maximize productivity for internal and external developers. Many organizations now manage APIs as products with full lifecycles and long-term roadmaps that are always evolving to meet business needs. We’ve observed that organizations that treat APIs as products—not projects—are more likely to realize the potential value of APIs as business accelerators.



We are starting to see more enterprises adopt this API product mindset. Of the people who have taken the survey provided in Apigee Compass, 24 percent identify APIs as products for software developers and 19 percent characterize them as strategic assets for creating business value. These are the perspectives we encourage you to take. This book aims to give those responsible for API programs the strategic guidance and tools they need to do just that—to establish an “API product mindset.”

How API products provide ongoing leverage

An API product mindset means designing and delivering APIs for long-term value at scale, and evolving them over time to meet changing customer needs. This is in contrast to approaching APIs as one-time projects, or several discrete projects, where they deliver more limited value in terms of extensibility, longevity, and reach.

A key part of an API product mindset is designing APIs like products. How an API is designed can dictate how easily it can be consumed by developers and thus how easily it can be leveraged in new ways in the future. If the API is designed only to build a connection within the scope of a project, its creator might neglect documentation, consistent design standards, considerations for versioning and security, and other factors important for future use and extensibility.

In contrast, if API designers operate with a product mindset, they'll prioritize ease of consumption and seek to increase the likelihood the API can continue to provide strategic value and extensibility in the future. Given that few things unite business leaders like the fear of being locked into a particular use case, strategy or business model, the significance of this distinction should be self-evident.



Organizations accustomed to building APIs within projects might face a range of challenges as they adjust to building and managing API products. In many enterprises, individual project teams may select their own software libraries and other implementation details—and if the APIs are exposed in a way that creates dependencies, unintended complexity and costs may arise if the API needs to be modified later.

Building APIs to meet specific project requirements is also common. This approach can limit the APIs' extensibility, either because back-end implementation details have become baked in (which makes updates more difficult and may make the API harder for developers to work with) or because

the APIs will require significant modifications to meet new use cases outside the original, narrowly defined scope. When APIs are not designed for easy, consistent, and intuitive developer consumption, the threat of complications and lost productivity looms large.

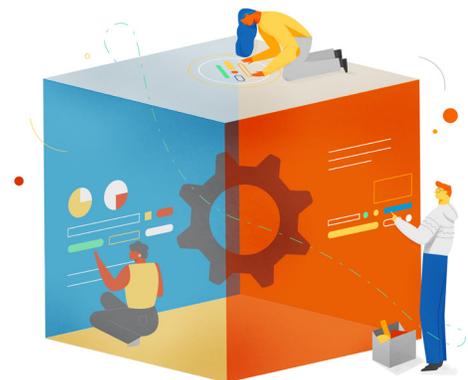
Suppose, for example, an undocumented API's payload uses a data element named "location" but actually returns a shipping address. To the original developer working within their narrow project scope, this naming convention might have made sense—but because of the lack of documentation and clear naming, future developers who attempt to leverage the API may become confused.

An organization's tendency to build APIs within projects is often a vestige of practices from the legacy days of monolithic apps, when software development was more expensive and the IT focus was on locked-down systems rather than agile adaptation, lightly coupled architectures, and scale. These are the characteristics required to operate at the speed of modern business, with the needs of internal development teams and external developer communities in mind.

API products should be designed in a consistent way that developers can easily consume, with both security considerations, such as OAuth protection, and usability concerns, such as documentation and sample code, given due urgency. An API should not be *over-designed*, however, which tends to limit future flexibility. Recall that service-oriented architecture (SOA) apps were also focused on reuse but have been largely supplanted by APIs in part because many SOA apps became too bloated—full of too much code that prematurely defined future uses.

When designing an API with ease of consumption or future use in mind, the goal is not to design for all needs but rather to start with an extensible base that presents the basic value proposition to developers and is easy for them to consume, and to then enable their feedback to guide future iterations.

Most importantly, the API product mindset requires that an API actually be managed like a product. It should be owned by a product manager, like any other product, and not by a project manager whose responsibility is to deliver on a list of requirements. The product manager should understand the customer requirements for the product and take responsibility for translating them into product requirements and a roadmap of iterations. To satisfy these responsibilities, the product manager should be aligned with other business and technical leaders on the goals and potential value of the API program. They should also have access to API management tools that provide visibility into how and by whom the API is being used and otherwise help to meet service-level agreements (SLAs), derive insights, and ensure the product is meeting customer needs.

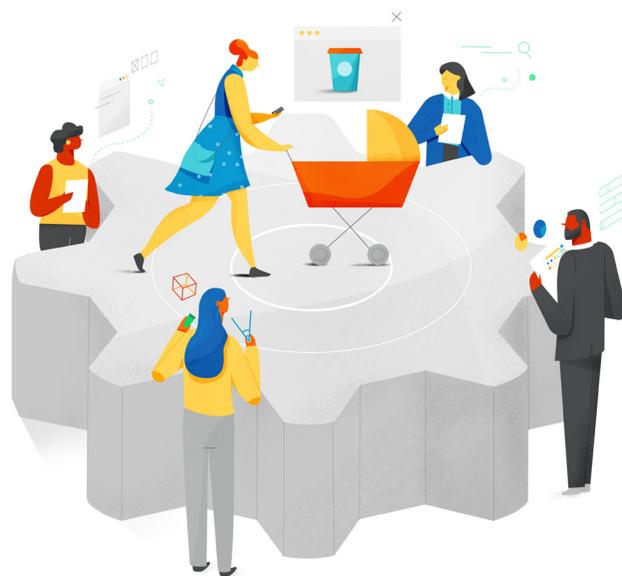


The product mindset pillars: outside-in, time to market, and iteration

Maximizing the value of API products involves not just technology but also the operational lens through which the technology is viewed. This right mindset rests on three broad pillars: customer obsession fueled by an “outside-in” approach, a focus on improving time to market, and a culture of constant learning and iteration.

Take an outside-in approach

At its core, an outside-in approach to building an API is about an obsession with the customer—i.e., relying on customer needs and preferences to guide product strategy. The outside-in perspective is data-driven, avoids making presumptions about customers, and considers both developers—the API’s direct customers—and the end users those developers serve. An inside-out approach, in contrast, tends to either base strategies on the resources IT says are already available or on an internal presupposition about what developers and end users might want or need.



The product mindset recognizes that a product’s future may evolve once the product interacts with users. Ahead of production, the product manager should understand the developer pain points the API aims to relieve or the opportunities the API will offer, but the full spectrum of use cases for the API may not be obvious at launch.

Mapping APIs weren’t necessarily created with ridesharing services in mind—that use case came later, through cycles of developers creatively leveraging the APIs and API providers responsively iterating their products.

With the right data—such as which APIs are being adopted, which APIs are generating the most calls, or which APIs are being most widely leveraged in apps—an API product team can understand customer behaviors and leverage learnings derived from those behaviors to iterate intelligently. A successful API product can be iterated into an entirely new way of doing business—but only if the provider has the right understanding of customer behavior to drive adoption and guide ongoing development.



Embracing the outside-in product mindset, then, means embracing new definitions of success. While a project can be considered “successfully delivered” even if it generates limited or no business impact, a product’s success or failure starts with whether or not it is adopted by users.



CUSTOMER CASE STUDY

AccuWeather creates API products to generate new customers, grow revenue

AccuWeather is the world’s leading weather media and big data company—serving over two billion users and handling billions of API data requests daily. AccuWeather’s JSON-formatted, easily cached, and highly efficient APIs are built to handle the enormous amounts of data required in a world increasingly ruled by new device types.

The company uses APIs to share its weather data with an array of partners across the globe and across a variety of use cases—like weather apps that live on smartphones, smart TVs, wearables, and in smart homes and connected cars.

Recently, the company focused on delivering API products with the goal of giving external developers and small and medium businesses a simple and fast way to build with AccuWeather’s services.

Taking the outside-in approach to productizing their APIs, AccuWeather realized that not all developers want to use the data in the same way. Some apps might need near real-time weather information—like severe weather alerts, current conditions, images from radars and satellites, and up-to-the-minute forecasts.

Others might need daily weather digests or forecasts that generally produce lighter API traffic

load. So, via the [AccuWeather API Developer Portal](#), the company provides tailored API products that let developers purchase and use according to their needs. Various weather services are offered as distinct APIs and include flexible, usage-based billing.

AccuWeather’s portal further supports developer consumption by providing detailed documentation and ongoing support for all of its APIs. Just ten months after the portal launched, and with limited marketing efforts, Accuweather attracted 24,000 developers who generated thousands of API keys and drove hundreds of purchases of Accuweather’s API product packages.



“We want to reach a new audience, and help them grow their ideas with AccuWeather. A single developer always has the potential to be working on the next big thing, and become our next big enterprise partner. We needed a way to reach them, and the AccuWeather API Developer Portal, powered by Apigee, accomplishes that.”

Mark Iannelli, Senior Technical Account Manager at AccuWeather

Adoption is only a starting point, though. The outside-in product mindset demands data—lots of it. Without data, an API product manager can't measure how the API is being used, who is using it, and whether users value it. The product manager can't discern if developers are using the API in unexpected ways. They can't guard against bots and malicious activity. Put simply, a product manager can't manage what they can't measure.

In many organizations, leaders may be accustomed to traditional project-oriented IT statistics instead of measures that tie API usage to business key performance indicators (KPIs), which can make communicating the performance and value of the API program a challenge.

“When we started out with our API program, we were looking at how many APIs we were producing and how many developers were registering on our portal. I think we've learned that it wasn't so much how many things we got out but the quality of the metrics,” said **David Freeman**, general manager of API enablement at leading Australian telecom Telstra, a Google customer.

“Now we're looking at things like revenue because our program is mature and we're monetizing. We're looking at the amount of developers that sign up and the amount of reuse of those APIs and how they're being taken up,” he added.

To define the right metrics, here are some of the typical questions that an API product manager with a customer-first, outside-in focus would ask:

- **“How often is the API called?”** This is a measure of how often a developer leverages the API and can indicate often the business's value proposition is presented to users.
- **“What kinds of calls are being made?”** Different API calls have different value for both the API provider and its consumers. For example, if the value measure is direct monetization, a call to

the provider's API to retrieve product information is less valuable than one that actually places an order for the provider's goods or services.

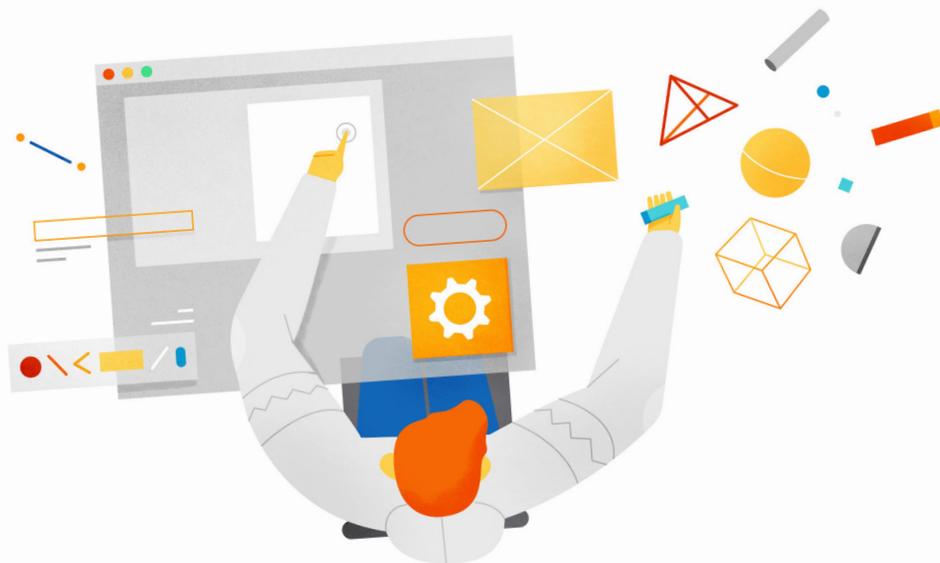
- **“Are developers changing their behavior?”** Developer consumption metrics can provide insight into the periods when API users are most active and whether engagement is increasing or decreasing.

Beyond hard-and-fast analytics, customer obsession should also include a variety of other means for interacting with and collecting feedback from users, such as participating in developer conferences, establishing a presence on popular message boards, creating a portal to facilitate developer access to APIs, and assigning a dedicated resource to manage developer relations.

Improve time to market with MVPs

Product-minded companies assume it is usually best to get to market (or in the case of an internal API, into production) early with a minimum viable product (MVP), and then to rapidly iterate based on developer feedback. Adopting a product mindset enables an organization to experiment with more ideas more widely and more quickly—and, as data comes in, to invest where products show promise and consider remediation where they don't.

Google customer Pitney Bowes, for instance, has adopted a product mindset to build a platform of hundreds of internal and external APIs. It has leveraged this platform and cloud technologies to both deliver its new cloud-based products four times faster and increase the number of developers working with its APIs by orders of magnitude.



Similarly fueled by a product mindset, Brazilian retailer Magazine Luiza has leveraged APIs, microservices, and hybrid cloud technologies to increase its software delivery speed from eight

deployments per month to more than 40 per day. Because of the increase in speed and agility, the company has been able to re-platform its mobile and e-commerce sites and introduce dozens of internal apps.

These apps include an Uber-like delivery app that has helped get orders to customers more quickly and an app for in-store employees that has helped reduce the time customers spend in store before making a purchase by 90 percent, to just four minutes on average.

Improving time to market involves many dimensions, including not only API design but also operational alignment. The point of releasing MVPs is to iterate quickly to maintain and improve them—which requires a certain amount of organizational coordination.

For example, an enterprise accustomed to SOA-era governance may find itself challenged by API-first approaches, which typically rely on modern, internet-era development techniques that resist heavy centralized management. To fully leverage its APIs, this enterprise should unify around lightweight governance processes including standardized documentation and reliable, consistent security and access-control mechanisms.

An API-driven business may move far faster than a traditional enterprise; after all, one of the advantages of implementing an API management layer is to decouple partner onboarding and new, rapidly evolving apps from relatively lethargic and brittle enterprise systems.

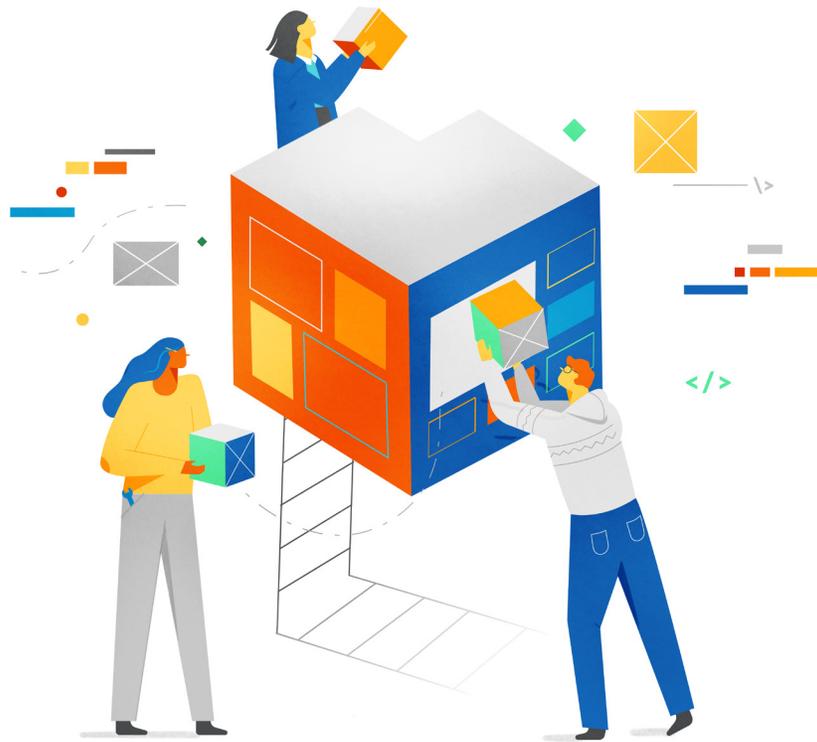
But a business doesn't move faster simply because it builds an API. In many enterprises, the annual project roadmap isn't prepared to bring on partners at internet scale. Sales, marketing, finance, and other groups may also be used to legacy cadences. Build and release processes, version control, and measurement and reporting should all be part of the vision for iterating API products so that developers can rapidly unlock new business opportunities.

The API team's roles and responsibilities should bring together all of these groups, aligning them under the speed APIs can provide and the business opportunities they can unlock. The team must communicate with both internal and external developers and business units at one end and with back-end IT teams at the other. It should generate a steady stream of new product ideas, driven by quickly introducing ideas to market, collecting data, and iterating.

“Before, there were very heavy processes. It would take two months to get all the information,” said Magazine Luiza CTO Andre Fatala, describing the period before the organization adopted API products company-wide. Now, he said, the company invests less in heavy governance and instead “spends more money on people that are going to make things happen.”

Mature products through iteration

Via an ongoing process of forming hypotheses, testing them, and unpacking discoveries, a product-minded API provider can gain insights to groom a backlog of updates and make decisions to scale up or shut down features or products. API providers should keep pace with developers' preferences and needs and provide customers with increasingly more useful and intuitive experiences.



By contrast, projects typically have a start date and an end date. If an organization needs to iterate on whatever the project delivered, it typically has to spin up a team, seek new funding, and, in many ways, restart the initiative. This is a cumbersome way of operating. Companies that embrace the product mindset find they can both deliver to market and iterate faster because they planned from the beginning to have an MVP, ongoing operational processes, and a product manager who is responsible for measuring and driving the product's success.

API programs require dedicated people and funding to sustain this process of ongoing management and iteration. To secure this funding, a product manager must be adept at using metrics to evangelize the value of the API program.

“The best way for us to be able to get over that hurdle of convincing the business to invest is linking the API program back to the business’s strategy,” Telstra’s Freeman said.

A guide to product management for APIs

How do you help instill the outside-in way of thinking across an entire organization for building and managing APIs? How do you operationalize rapid delivery of API products that empower developers? How do you create an API program that evolves to meet customer needs and holds strong in a world with ever-changing innovations and fierce competition? Answering these questions starts with the API team.

Introducing the API team and the API product manager

The API team manages APIs across their entire API lifecycle, including productization and, in some cases, monetization. API product managers are at the heart of managing, supporting, and optimizing the team's workflows—they are the engine that keeps the process of new releases, iteration, and customer obsession running. This requires understanding and implementing best practices and recommendations across each stage of the API lifecycle.



Many enterprises might not have an established API team or even an API product manager—even if they are already developing APIs. At some organizations, it might not make sense to have a centralized team. In some enterprises, the makeup of the API team and the enumeration of responsibilities may change as the business evolves, starting with less centralization and gradually shifting into a dedicated unit with more precisely defined roles.

For some, this might be the first time they've heard the concept of an API team and the role of the API product manager.

No matter where you stand, it's critical to find and organize the resources needed to execute on the roles and responsibilities that live within the API team. Whether that manifests as an organizational unit or a loosely coupled group of people from across different teams, the team is responsible for understanding the needs of customers—and helping the enterprise deliver on the promise of developing API products that will provide developers with the tools they need to create breakthrough products and experiences for end users.



The core roles and responsibilities of the API team



The API Product Manager: The API product manager is responsible for putting the cross-functional conversations and activities in place to help an organization develop its vision for API products. The manager turns this vision into action by managing each stage of the API product lifecycle, prioritized by API consumption data and other feedback from API consumers and customers. The product manager is the subject matter expert of the API product, deeply understands its technical specifications and benefits, and can effectively communicate those benefits to others—including colleagues and executives on the business side of the enterprise. Depending on the organization, their responsibilities may include running scrum meetings, prioritizing a features backlog, defining the right KPIs, preparing partner presentations, or helping launch the API into wider ecosystems.



The API Architect: Like the architect that drafts the plans for a house, the API architect is responsible for planning, designing, and reviewing the construction of APIs. They guide the creation of APIs and the development and testing required to meet the highest standards in API design. Architects should clearly understand the difference between API products designed for consumption and API products designed for exposure. They might be responsible for understanding the details of underlying systems or liaising with back-end teams that do. Their work should reflect an understanding for how it affects other members of the API management team as well as the developers consuming the API.



The API Developer: If the architect creates the plans and drafts for the construction of the API, the API developer is the actual builder—the person who brings the wants and needs of the API consumer or application developer to fruition. Their goal is to produce intuitive and highly consumable APIs. They fulfill this goal not only by building the API itself but also by contributing to resources that will help developers most effectively leverage the API, such as documentation. The API developer should be an expert web developer, well-versed in both consumption- and exposure-oriented API development. They should have a strong understanding of the ins and outs of the API management platform to implement security policies, traffic management, and other protocols that support the scaling of APIs per standards consistent across the organization.



The API Evangelist: Like any product evangelist, the API evangelist is the voice of API consumers. The evangelist should deeply understand developers and ensure they have all the things they need to be successful. This role is responsible for the vital task of managing the developer portal and ensuring developers have access to detailed information on the product offering, including documentation and software development kits (SDKs). They are responsible for internal and external developer outreach. This outreach might include not only answering questions, running hackathons, and bringing developer feedback to the API team to inform and support product roadmap discussions, but also marketing the APIs via SEO, targeted ads, and other methods. Evangelists can play a crucial role in attracting partners to the company's APIs and thus in expanding the ecosystem of participants leveraging the company's offering. Great API evangelists have a passion for seeing their team's work blossom into the applications that serve the end user.



The API Champion: The API champion is to the executive boardroom or the lines of business what the API evangelist is to developers. They connect an organization's API programs to the business value they provide by translating data points that may only make sense to technical leaders into metrics that track directly to important company goals—like increasing customer satisfaction. They enlist strong support from executive sponsors who have the ability to provide the funding and resources API teams need to be successful. They should be analytical thinkers and powerful influencers that can develop strategic business goals while paying attention to the technical capabilities of an API product offering. They should always understand how the business benefits of API products apply to app developers and customers as well. They should ensure the team continues to move in an agile fashion, so the organization can get more MVPs to market and open the door to possible new lines of business.

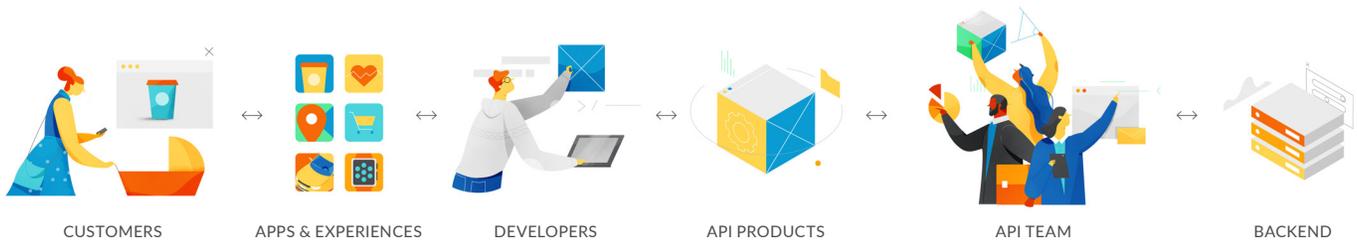
Though described here as discrete roles, each of the above personas might not manifest in one person. For example, at some organizations, the duties of the API product manager also encompass the responsibilities of the API champion. At other companies, it's better for the product manager to focus on the technical aspect of API products while the API champion focuses on communicating business value.



One of an API team's first steps is to bring uniformity to the API creation process by ensuring that APIs are created for consumption and that access to systems of record and other back-end data is provided in a

consistent way. The API team should think of itself as a refinery that sits between the raw materials housed in back-end systems and the API products that present those raw materials as polished interfaces that developers can easily use.

When you layer a mature API product mindset on top of the API team's core responsibilities, the graphic above grows into the full "Digital Value Chain" represented below. The chain emphasizes the two key customers whose experiences the API team directly impacts and who should form the focus of the API team's obsession with customers: developers and the end users for whom developers create apps and experiences.

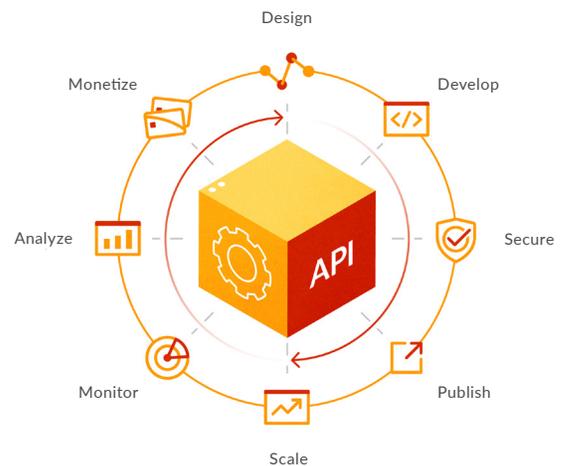


To create an effective value chain, the API team should focus on three practical objectives: building a minimum viable product, curating a world-class developer experience, and championing the business value of APIs.

Building a minimum viable product: design, develop, secure

Full lifecycle API management involves all of the things it takes to get an API up and running and into the hands of developers, so that they can both begin innovating and start providing feedback to inform how the API is iterated upon over time.

Though the numerous stages of the lifecycle may suggest a lengthy process, remember: speed is the goal. The lifecycle should focus on releasing MVPs that present the core value proposition to developers and can be rapidly scaled or improved based on user feedback.



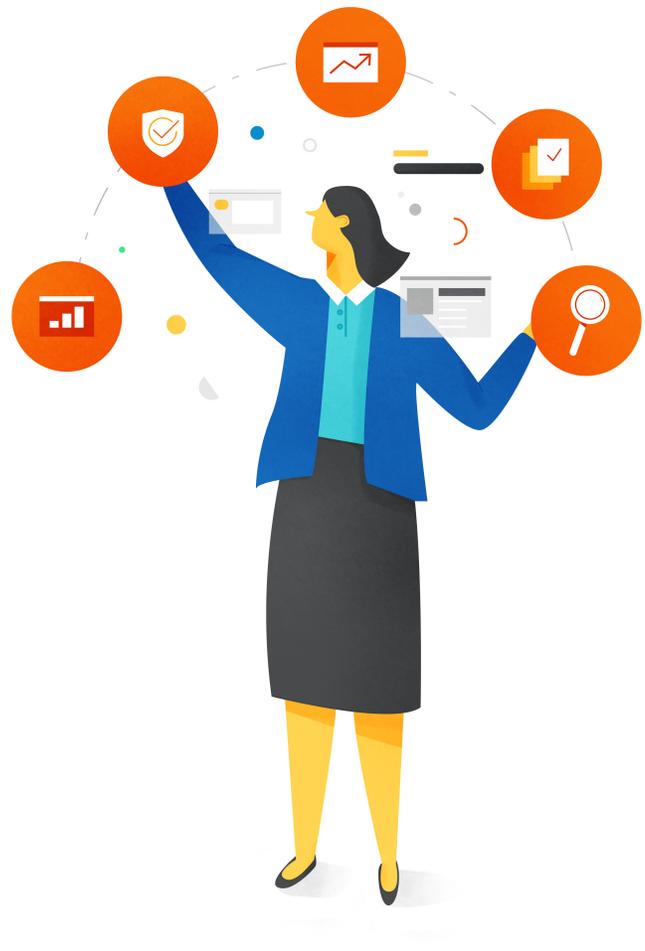
API teams should not expect all MVPs to succeed. The point is to launch more releases than would have been possible using legacy approaches and to quickly realize which ones are gaining adoption and which experiments justify additional investment. Some experiments will fail, but those that succeed may represent not only new products but gateways to entirely new business models.

When managing the lifecycle of these MVPs, keep these principles in mind:

- **Build APIs developers love and that are easy to use:** Design according to RESTful best practices, focused on consumption and consistency, and provide a clear statement of the value proposition the API represents to developers. Avoid premature optimization and hide unnecessary complexity from developers. Remember, developers likely do not have time to learn the inner workings of your system and should not need to: they simply want the data they need to enrich their application.

- **Protect your APIs:** Enforce a consistent set of security policies and protocols across all APIs—private and public. Employ authentication like OAuth/OpenID Connect in conjunction with transport layer encryption (TLS) to protect data and control who accesses it, and consider spike arrests and per-app usage quotas to help maintain API performance.

- **Test and deploy APIs:** Sync the API lifecycle with a modern, agile, iterative software development lifecycle (SDLC) and automate API testing and deployment.



CUSTOMER CASE STUDY

Pitney Bowes GeoSearch API improves satisfaction for SendPro® customers

After nearly a hundred years in the mailing and shipping solutions industry, Apigee user Pitney Bowes has transformed into a provider of digital logistics and e-commerce solutions through the power of APIs and the cloud.

Customers of **SendPro**[®], an online SaaS application and a series of Android-enabled devices built to help streamline office mailing and package shipping, had been dealing with a returned mail issue that the solution's engineers were trying to solve. Users were getting mail returned due to a flaw in the application that was allowing manually entered, incorrect addresses to continue to be processed onto postage and shipping labels.



Another team inside Pitney Bowes working on location APIs heard about the problem and suggested the **SendPro**[®] team consider using a fully available product API called **GeoSearch**. **GeoSearch** is a type-ahead API that auto-completes address entries and only suggests known and valid addresses. It reduces abandonment, increases conversions, and reduces and eliminates returns.

In just one week, the **SendPro**[®] product team integrated the **GeoSearch** API into their product. And after two weeks of required testing, the feature hit the market. The **SendPro**[®] team was able to increase speed to market and eliminate the issue quickly.

This is a great example of how an API that was designed for customers to build upon can also be put to meaningful use internally: while Pitney Bowes customers already used the **GeoSearch** API in retail and e-commerce, now it's used internally to help improve the **SendPro**[®] customer experience and solve a critical business problem.

*“Our API products are helping our customers enhance and enrich their applications, business processes, and workflows, and solving critical business problems. We are now also enhancing and enriching our own applications, business processes, and workflows with our own API products like **GeoSearch**.”*

Today, our location API products are ingredients in over a dozen other Pitney Bowes SaaS products and internal business processes and workflows spanning multiple lines of business.”

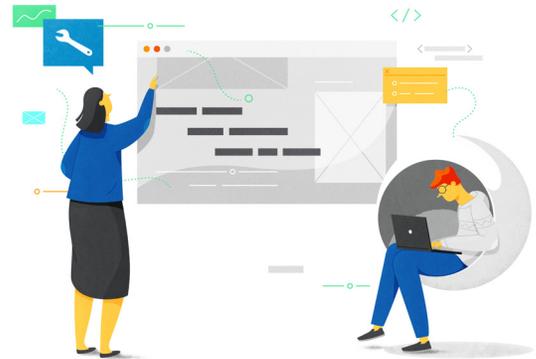
Jon Spinney, Director of Product Management at Pitney Bowes

Curating a world-class developer experience: publish, evangelize, scale, monitor

The API lifecycle starts with designing and releasing top-notch MVP API products. The next phases are about driving adoption; monitoring and analyzing consumption patterns and other forms of developer feedback; ensuring performance; and beginning to plan iteration. API teams can prime themselves for success across these lifecycle phases by focusing in three areas:

- **Establish a developer portal to drive adoption:**

The faster a developer can go from accessing an API to creating a new experience or service, the more information an API team can gain about its customers and new business opportunities. Businesses can significantly improve time to market by investing in a developer portal that facilitates easy access to well-documented APIs, testing tools, blog posts, forums, and other features to encourage easy consumption. The developer portal is “the primary interaction point where somebody can go from awareness to activation to acquisition for an API and be able to bring it up to a ‘hello world’ within maybe 30 minutes,” said **Thomas Squeo**, senior vice president at Google customer West Corp, a telecom firm.



- **Monitor performance:** Developers’ apps and services only work if the underlying APIs do—so it’s vital to monitor performance, ensure that SLAs are being maintained, and protect against abuse. Proactive monitoring of API traffic also conditions the team to focus on measures of consumption rather than measures of completion—and to start identifying problems and wins and charting possible paths for the API’s iteration.

- **Invest in API evangelism:** In the world of API products, there is no “if you build it, they will come.” For external APIs, the API team needs to establish a presence in the communities where developers already congregate, whether that’s forums, meetup groups, or conferences. The team should support its APIs with marketing and promotion, whether it’s targeted ads buys or webinars with influencers or thought leadership content. For all APIs, including internal releases, the team



needs to sell developers on adoption, which means making the value proposition clear, providing tools that make the API easy to use and experiment with, and creating feedback loops with API consumers. Developers are the API team’s direct customer, so the team should expend significant effort understanding developer needs and catering to their preferences.

Championing the business value: establish KPIs, iterate, monetize

An API's first lifecycle ends with iteration and should be accompanied by a maturing ability to communicate the API's business value and an API roadmap that evolves in response to developer feedback. This process involves several API team competencies:

- **Define the right KPIs:** To turn API consumption data into actionable insights or persuasive measures of the team's value, the team must define KPIs that connect consumption to revenue, customer growth or some other core business KPI.
- **Manage iteration:** Once an API has been in the wild with developers for a short period, the team should assess data to plan iterations. Is performance suffering because too many calls are involved? Do developers require additional features to make better use of the API product? If a team is managing its first API and funding has already been secured, the process of addressing these questions may be somewhat straightforward. For more mature programs juggling lifecycles for multiple API products, establishing the right feature backlog processes and development cadence may be crucial.
- **Consider monetization:** Many organizations drive adoption by releasing APIs that give developers basic or full functionality at no cost. APIs that allow access to particularly valuable data or functions, however, may be good candidates for monetization.



What's next?

You might be assembling your organization's first API team or working to optimize a team that already exists—either way, a great way to move forward is to identify and scrutinize the characteristics of project and product mindsets. Then, use your understanding of the product mindset to align collaborators, distribute responsibilities, and develop a culture of fast, customer-obsessed iteration.

	Project Mindset	vs	Product Mindset
Perspective	Inside-Out		Outside-In
Goals	Completion		Consumption
Metrics	Completion, Cost, Acceptance		Consumption, Revenue, Customer Satisfaction
Requirements	Exhaustive		Iterative
Teams	Intermittent		Continuous

At many of the most successful companies, leaders throughout the business—not just technical leaders—understand the value proposition of APIs. However, the API team might struggle to garner the resources it needs, let alone to drive API adoption, if leaders don't understand and throw strong support behind the team's efforts. To increase support among leaders, remember the power of KPIs that show how API consumption has impacted customer satisfaction, revenue, or growth.



As your team matures, consider broadening its horizons to new use cases that can unlock new business opportunities. If you've been focused on internal APIs that accelerate development of new products, you might expose an API to harness innovation by external developers. Some teams might consider turning API products into a revenue stream and packaging them into monetizable bundles that serve particular developer needs.

Virtually all businesses want to move faster. Most have systems and functions they want to use more efficiently. Many have data full of value just waiting to be unlocked. The vast majority want to accelerate partner participation so they can focus on what they do best while relying on others to help fill go-to-market gaps and provide scale—and so they can better serve their customers. APIs designed, delivered, and managed as products can help with all of these things. They can make a company's valuable assets and capabilities available for developer creativity. They can eliminate redundancies and accelerate development. API products can open an organization to new partners and ecosystems. And they can do all of these things fast. The journey starts with getting the first API product on the roadmap.

Whatever path an organization's APIs take, the point is this: because APIs are increasingly how business gets done, they've become a fixture not only in IT conversations, but also in the boardroom. Enterprises that approach APIs strategically—with a product mindset—are poised to thrive in today's economy.

Now that you've finished reading, why stop learning?



API PRODUCTS MICROSITE

Take a deeper dive into the API product mindset concepts and strategies you've learned by exploring a microsite full of success stories, videos, eBooks, articles, and more.

KEEP LEARNING

FREE TRIAL

Explore Apigee Edge, a full lifecycle API management platform that helps you manage the entire API lifecycle from design through iteration—so you can get your API products into the hands of developers today.

TRY IT FREE

THE DIGITAL TRANSFORMATION JOURNEY

Learn how to accelerate your company's digital transformation with 10 ingredients, based on lessons learned from Apigee's work with hundred of companies on their digital transformation projects including Walgreens, Pitney Bowes, and Experian.

DOWNLOAD EBOOK



apigee

Share this eBook

on social



with a colleague



Google Cloud

© 2018 Google Inc.