STATE OF API INTEGRATION
2018 REPORT

cloud elements | OPENAPI INITIATIVE

Table of contents

A word from our CEO

Our Mission is to Integrate App Ecosystems by Unifying the World's APIs.

Integration. It's that pesky part of making a product that no company actually wants to do. (And who can blame them? It's not easy.) But it's also the part that really has to be done well. Because if your product can't be used by the people who need it, then why build it in the first place?

It goes far beyond a simple connection to an API. Cloud Elements has re-imagined the world of application integration so you and your customers can focus on the data they care about.

This report addresses the continued growth in API adoption, where the market is headed with ease of integration, and new trends in the coming year.

Happy Integrating.

Mark Geene
CEO - Cloud Elements
@mgeene

cloud elements

Meet our Contributors

Ross Garrett
at Cloud Elements

@gssor

Ross Garrett is the VP of Marketing at Cloud Elements - responsible for market strategy, product positioning and evangelism. He is a well-known speaker at developer events and other industry conferences.

Kin Lane
at API Evangelist

@kinlane

I am the API Evangelist. Not in the sense that I'm evangelizing a single API to you--In the sense that APIs are important for everyone to be aware of. I'm paying attention to not just the technical, but the business and politics of the web API movement.

Isabelle Mauny
co-founder and CTO of 42Crunch

@isamauny

Isabelle Mauny, co-founder and CTO of 42Crunch spent most of her career at IBM, across a variety of technical roles, at the European level. She was part of the IBM WebSphere Strategy board and played a key role in the deployment in Europe of flagship products such as the WebSphere Application Server or DataPower appliances.

Mark Boyd
API industry analyst

@mgboydcom

Mark Boyd is a writer and API industry analyst, who writes regularly for ProgrammableWeb, and The New Stack. He has chaired several API conferences, and published a number of ebooks on the API lifecycle, best practices for API adoption, and API uptake in industry verticals.

SECTION 1

Introduction

SECTION 1

The State of API Integrations

As we learned in last year's State of API Integration Report, the fast-paced world of SaaS Applications and app development is nothing without advanced API integration. Not only have APIs become a necessity for application customers to streamline operations across their business and product suite, they have become an integral part of product development, business strategy, and scalability. As a result, customers will continue to seek out applications and APIs that can be easily integrated.

Not only have APIs become a necessity for application customers to streamline operations across their business and product suite, they have become an integral part of product development, business strategy, and scalability.

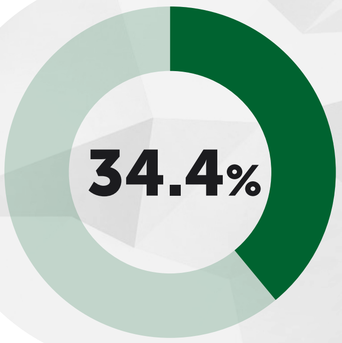
It's no longer sufficient to offer a standalone, isolated product—developers must build flexible platforms that not only offer highly functional APIs, but also integrate seamlessly to the ecosystem of products and services used by their customers. APIs are the foundation for this network effect, and must continue to evolve to meet the expectations of developers. The role APIs and API-based integration need to play in creating a customer-centric platform is a key theme we discovered in building this year's report.

The proliferation of applications and subsequent API-based integration demands that developers now consume and build against a multitude of endpoints, languages and platforms. To develop a top-down view of this rapidly growing network, we surveyed hundreds of application developers for feedback on their own APIs and their experience with others they work with. We've distilled all of that information into the most noteworthy stats, changes and trends in our comprehensive State of API Integration 2018 Report.

Important Findings:
Building Platforms Instead of Products

This report will cover key findings from the survey we distributed to collect insights into the state of API integration. It will share our predictions on how these findings will change the industry in 2018 and beyond. We will explore prominent themes, including the expectation that APIs ease integration, rather than stifle it, and that successful applications should be more like platforms than products. For example, 36% of respondents stated that their primary motivation to become a platform provider was sparked by their desire to maintain 'stickiness' with customers by integrating easily across the ecosystem of applications.

While most companies "considered themselves a true platform provider," ***one-third*** of respondents said they offered pre-built integrations for non-technical users. Without such an option, customers with little-to-no development experience will struggle to build their own workable integration—limiting the network effect of that platform. This report helps define what truly constitutes a flexible 'platform,' and the implications that this definition has on the industry.



While most companies ***"considered themselves a true platform provider,"*** **one-third** of respondents said they offered pre-built integrations for non-technical users.

Our data also exposes the growing desire for innovation in event-driven integration:

*Nearly **40%** of those surveyed responded that their apps did not support event-based framework, but it was the number one answer when we asked which API technology area was most in need of innovation.*

While this architectural pattern is enjoying something of a resurgence - as it reduces coupling between services and provides a far more efficient way of synchronizing and updating data - work still needs to be done amongst API providers to embrace event-driven capabilities. Find even more of these key observations in the following sections!

Looking Ahead in the World of APIs

The report also gives us an opportunity to look forward to 2018 and beyond, and predict new trends in the API space, including:

- ✓ Connecting to an API, and truly integrating with an API are not the same - developers and API providers alike must consider all aspects of integration and find ways to standardize and simplify this process.
- ✓ Applications should focus on developing 'stickiness' and the ability to easily integrate to other products and partners in order to stay competitive in their markets and drive both new and renewed subscriptions.
- ✓ The continued growth of public APIs that are open to any developer will be paired with pre-built integrations that non-technical consumers can implement easily to streamline processes.
- ✓ Developments will be spurred in particular by verticals like FinTech, Banking, Healthcare and Human Capital Management.
- ✓ Further support for event-based integration, which is a feature commonly requested by developers, but that has commonly been unavailable for apps currently on the market.

This introduction serves only as a high-level overview of the results we collected from our State of API Integration survey. Keep reading to get the full breakdown on the current state of the API industry, a look at what's trending and why, and a look ahead to where we believe API integration is headed.

The API Landscape of 2018

Contributed Content from Kin Lane, API Evangelist

“

I’ve been watching the technology, business, and politics of APIs in a full time capacity since 2010, and the biggest trend I am seeing in 2018 is that it’s no longer a conversation about whether or not businesses should be doing APIs. In 2018, it is expected that you need APIs to do business in this digital age. The companies, organizations, institutions, and government agencies who are just beginning to invest in their API infrastructure are quickly realizing how far behind they are when it comes to the efficient delivery of data and content to web and mobile applications, as well as the ability to work with Internet-connected devices, and take advantage of the benefits of machine learning and artificial intelligence.

However, the platforms who have been investing in their API platforms for the last couple years are shifting their focus towards increased investment in event-driven architecture, and an API governance strategy to further optimize and maximize on their earlier investments. To be able to operate their platforms at scale, API providers have embraced a microservices-centered, robust lifecycle approach to delivering their APIs, with an OpenAPI-driven definition acting as a central contract. Providing a blueprint for designing, deploying, managing, and properly integrating, and extending each service to wherever it is needed, no matter where that might be. Providing the machine readable gears that are needed for the automation required as part of an ongoing organizational digital transformation.

In 2018, OpenAPI has become the contract for doing business with APIs, and is key to API providers efficiently delivering and managing their growing number of microservices. This contract is being used to manage the platform, but also ensure that services properly reach outwards to API consumers with up to date API documentation, robust SDKs, IDE integration, and API discovery solutions for developers. These contracts are also being used to drive spreadsheet integrations, Salesforce connectors, and the growing number of integration services that have emerged on the landscape to help non-developers take better advantage of API-driven resources. OpenAPI definitions are the technical, as well as business contract for mature API platform providers, allowing them to manage their APIs at scale, and reach a technical as well as business audience.

With machine readable contracts in play, driving an increasingly automated lifecycle, API platforms are looking to further refine their API governance strategy, helping ensure the continued delivery of highest quality APIs at scale. When this is combined with a heightened investment in real time streams of information, event-driven infrastructure, and the development of relevant machine learning models, API industry leaders are looking to shift their platforms into high gear.

This all makes the conversations around whether or not we should be doing APIs a distant memory, turning hard fought lessons around delivering, integrating, and aggregating APIs into the fuel for the next round of investment in event-driven architecture, machine learning and API governance that will help drive business growth for the next decade.

Behind the data

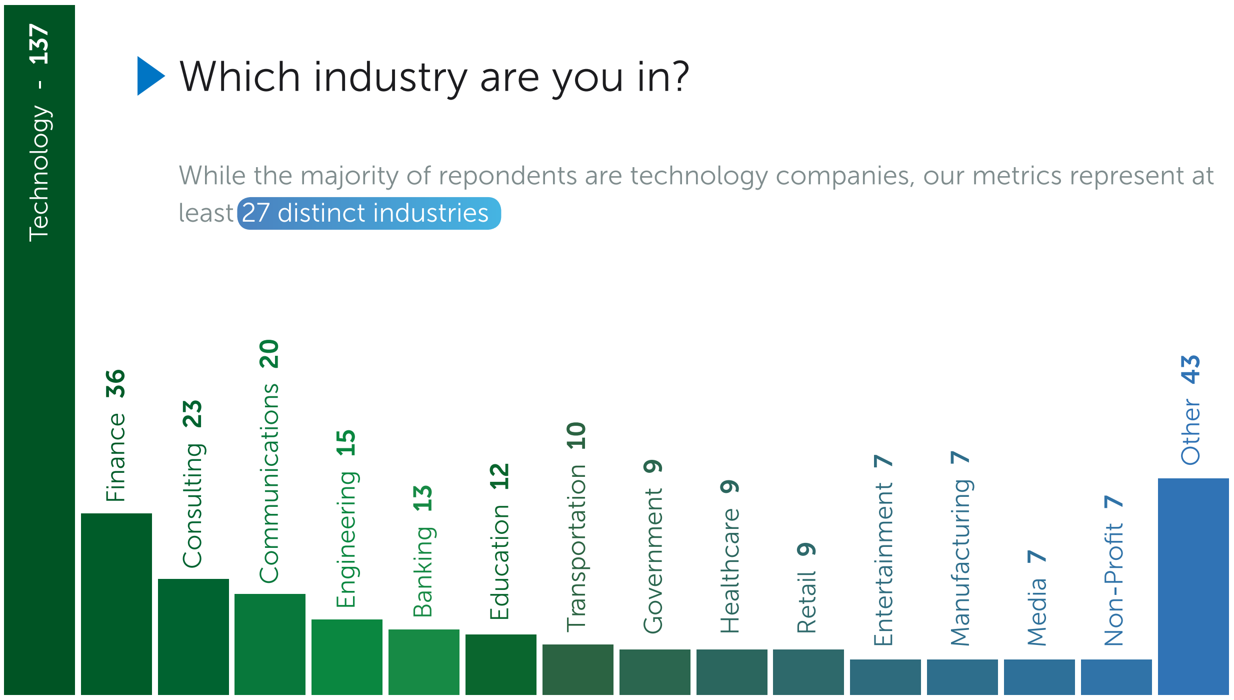
Last year, Cloud Elements released the first ever State of API Integration report, which analyzed data from within the Cloud Elements API Integration platform, including over 1.6 billion API calls to 160+ endpoints.

We are taking the 2018 report to the next level.

This year's report expands upon the data from the Cloud Elements platform, featuring data collected from 400 companies, in 27 distinct industries, with annual revenues ranging from less than $100,000 to over $25 billion. Collectively, these participating companies spent 200,000+ development days building over 4,500 API integrations in 2017. Our inaugural State of API Integration survey was distributed between September 2017 and March 2018 to representatives of these companies. Their responses contributed to the identified trends surrounding the existing demand for API integrations and the roadmap for leveraging integrations in the future.

Which industry are you in?

While the majority of repondents are technology companies, our metrics represent at least 27 distinct industries

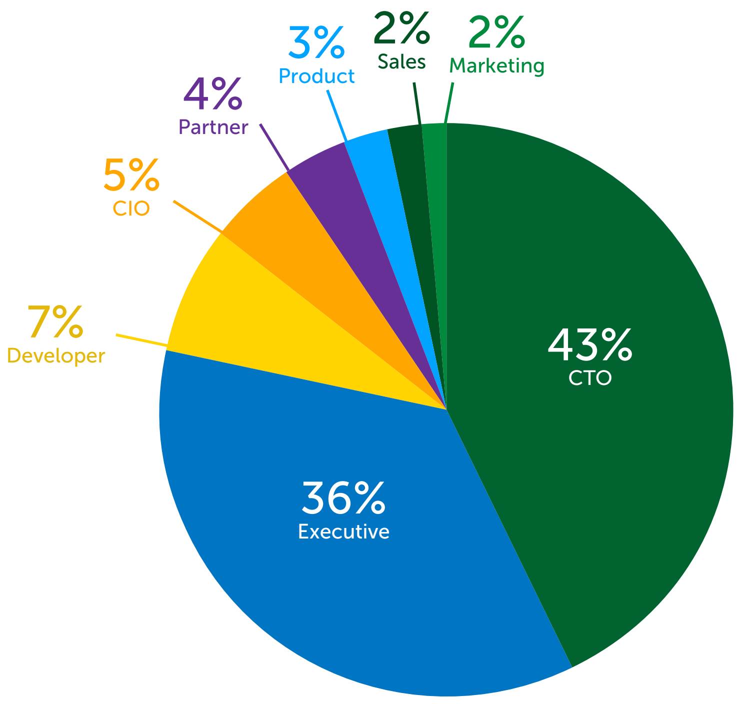


The report also includes industry expert insights from Ross Garrett, VP of Marketing at Cloud Elements, Kin Lane, The API Evangelist, Mark Boyd, writer for The New Stack and ProgrammableWeb, and Isabelle Mauny, co-founder and CTO of 42Crunch.

The 2018 State of API Integration Survey Respondents

Integrating the World of APIs

The analysis in the 2018 State of API Integration report represents data collected from companies across 44 countries and 6 continents



What is your position?

SECTION 2

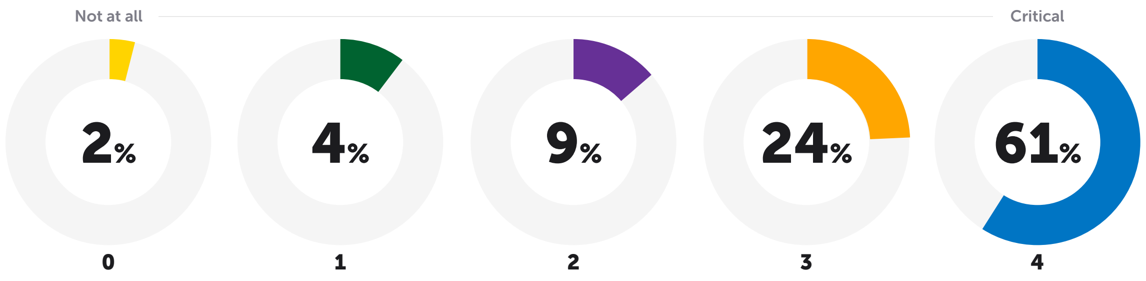
Trends in API Integration

SECTION 2

Trends in API Integration

As the number of public and private APIs grow month over month and year over year, it's no surprise that at least 85% of organizations we surveyed consider Web APIs and API-based integration fundamental to their business strategy and continued success. Of these, more than 60% have invested in a public facing API and the surrounding infrastructure to manage access for developers and partners.

Over 60% agree, API Integration is critical to their business strategy.



1

Building a Platform Strategy:
How to drive new revenue opportunities or deliver value through API integration.

2

Drivers of Adoption:
Changing how developers and end-users consume APIs and integration services.

3

Strategic Shift of Integration:
How API integration has matured to enterprise applications like ERP.

4

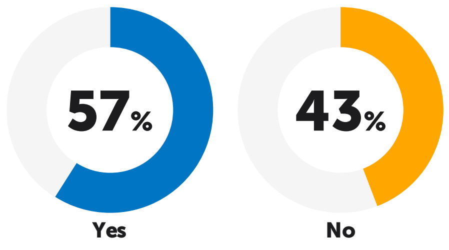
Evolution of B2B Integration:
Organizations are evolving the way they share and synchronize data with partners.

Building a Platform Strategy

Today's most successful software companies have made the leap from selling products to selling a platform. The appeal of such a move is understandable: software products generate only a single revenue stream, while platforms, by connecting different groups of users and services, can generate multiple revenue streams.

Our survey responses demonstrated that more than half of organizations today would consider themselves to offer a platform or to be on the path to offering a platform. Of those that don't, it's important to have a made a conscious decision in this regards - hope is not a strategy, and many product organizations are being disintermediated for moving too slowly in this regard.

Do you consider your organization a platform provider?



Platforms enable a multi-sided business model that can be a key driver of growth:

1. Single-sided business model limits your company's growth and longevity. You're simply delivering a product in the way that you always have, to the customers you always have, without really thinking about the network around your organization.

2. Multi-Sided Business Models take that a step further. In this model, a platform company enables other businesses to interact with your technology in order to build new services and applications.

3. Third party integration can create greater adhesion, stickiness and relevance across your market.

Model A
Basic Usage Metering

This is the most obvious method of generating revenue from your API—directly charging for it. This strategy works best if your organization provides access to data or services that people want to pay for.

dun & bradstreet

E.g. Dun & Bradstreet is a data provider, and expose data as a service via API. In this case, customers are willing to pay for programmatic access to D&B's services to enable new business processes and products around the D&B platform.

Model B
Upselling API Integration

SaaS providers use this model often, as customers have become increasingly demanding of integration capabilities for any product they buy. Adding API integration to a subscription offers a strong motivator to upgrade to a higher package, and creates a stickier relationship with customers by allowing end users to customize their experience and workflow more easily.

CONCUR

E.g. Concur provides pre-built integration options with their product, so that subscribers of the service can easily synchronize expense management data with ecosystem of apps used in their back-office processes—such as accounting and ERP systems.

Model C

APIs as a Product

APIs that are themselves the product can be the most complex monetization models, but they also often provide the highest value—collecting some revenue share or service-based fee for the product delivered via API.

PaySimple

E.g. Paysimple—a leading payment gateway provider—offers API access to their payment services. Customers of this service will then be charged a percentage of the payment amount for each API transaction.

Model D

Distribute Value through Partners

Organizations can scale the reach of their product by integrating their API with strategic partners. This has the potential to open your business to existing application ecosystems which you may not be able to reach alone.

sage

E.g. Sage works with various ISVs to distribute their products and services as part of a broader offering. This network effect has enabled them to achieve the scale and growth that would be difficult in a direct sales model.

Model E

Improve Operational Efficiency

API integration can help you build end-to-end solutions more efficiently. They help you innovate faster, engage customers and partners faster,and perhaps most importantly, they allow you to iterate (and even fail) faster. The time taken to build or automate business processes can easily improve from a couple months to a couple hours, solely because of the efficiency gained from an API integration.

MicroStrategy

E.g. Microstrategy—a leading business intelligence platform— provides pre-built integration to several leading enterprise SaaS applications, so their customers can quickly access and integrate the data they care about without needing to build anything from scratch.

While these are only examples, each describes a path towards revenue growth and ecosystem expansion with APIs. We expect that leading platform providers will leverage more than one (if not all) of these models as they grow.

Ultimately, a move toward "best-of-breed" is on the rise - meaning that every product, in every vertical, is better served by enabling integration to the ecosystem of apps around them rather than remaining a hermetically sealed environment. Over the coming year, we anticipate to see more of the world's successful software companies continue to make the leap from selling products to selling a platform.

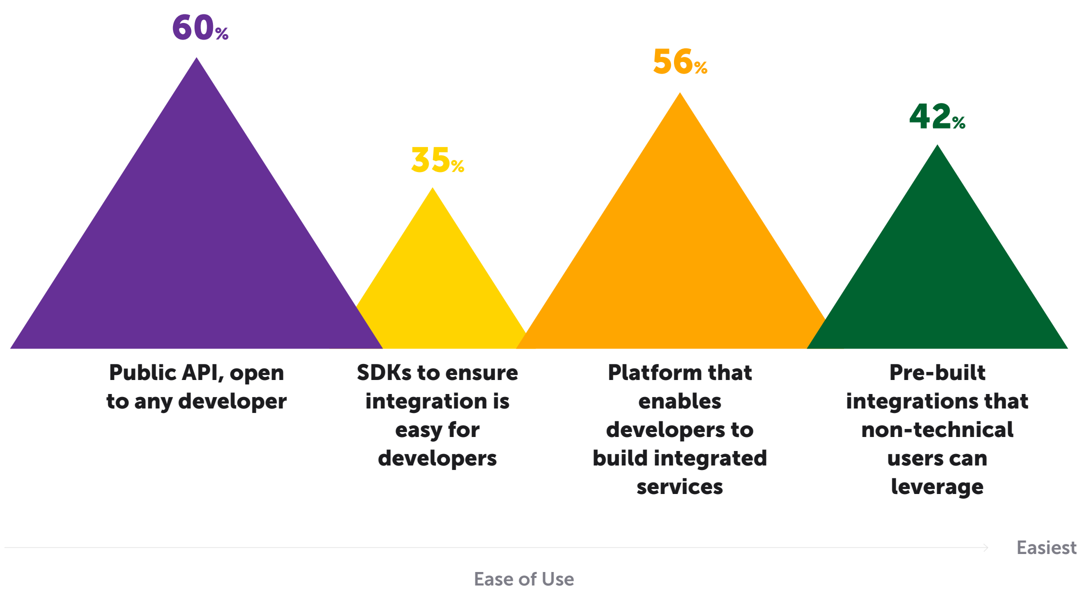
Drivers of Adoption

This report paints a picture of hockey stick growth and untold opportunity, but the growth in APIs hasn't come without challenges. Many application providers, businesses and even platform companies reported concerns around API adoption.

Issues with API adoption can be related to developer experience—API style, documentation, functionality, etc. But the real issue is far more systemic than DX. Adoption is limited by the addressable market, i.e. the number of developers that are available to learn, build against, or even care about your API!

How Organizations are Driving API Adoption Today

(Respondents selected all that apply)



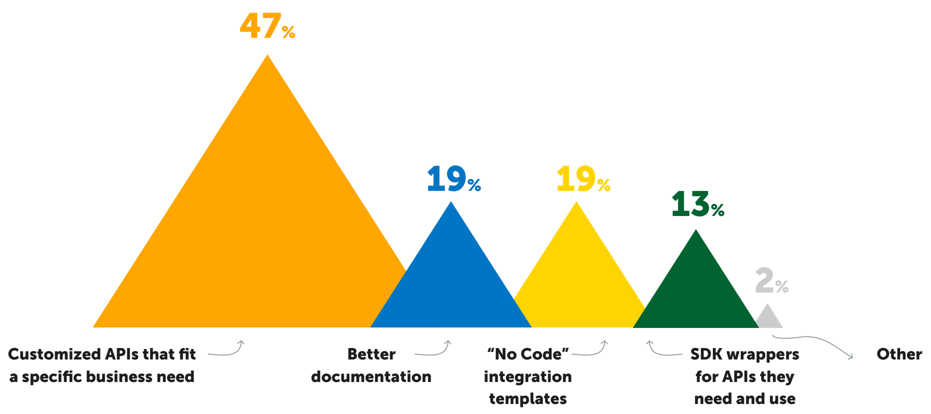
Our 2018 survey points to evolving and increasing need in the world of API integration, creating a possibility for non-technical users to build the workflows they need. By offering "no-code" or "low-code" capabilities, we can massively expand the total addressable market for APIs and instantly solve the common adoption concern.

Customized APIs

One Size Does Not Fit All

As API integration evolves and matures, it has become very clear that not all API consumers are created equal. Data objects may need to be modified based on the device type; orchestration or composition may be needed depending on the sophistication of the client; security might need to be adapted to fit mobile, web, or IoT scenarios. The list goes on.

What is the highest demand **from your customers and partners** for API integration?



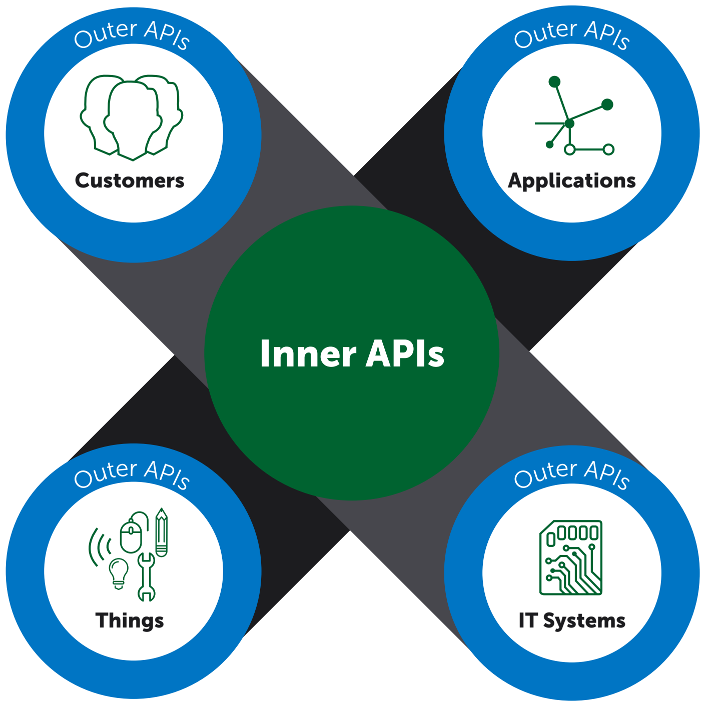
NETFLIX

A great example of this diversity comes from Netflix, where their API development team learned quickly that a “one-size-fits-all” approach to API integration wasn’t going to work. The team found that different client applications (such as a desktop browser, a mobile application or a smart television) required different functionality as they accessed the Netflix API, thus forcing the API team to build an increasing number of features to address the demands. Once continuously updating this single interface became unmanageable, they introduced an API experience, or mediation, layer on top of their existing API platform that allowed each UI team and even partners to optimize the API experience for their specific app or device.

With each new innovation in connected devices, and each new user and application experience that accompanies these devices, organizations must consider an application architecture and integration strategy that embodies flexibility and agility. API mediation is becoming a more familiar concept as many organizations begin to evolve beyond their first foray into the API Economy.

API mediation is a solution for enriching or personalizing interactions between distributed application and service components. We like to call this API Experience Management. The key to the API experience model is placing a new mediation or abstraction layer between API consumers (apps, services, and “things”) and API providers (services and applications). This layer wraps the backend API (the inner API) and exposes a personalized and managed API (the outer API) to each defined constituency of consumers, simplifying the developer experience while also ensuring loose coupling.

The new outer API may also aim to offer more advanced features to your developers, providing a facade on top of the inner API to enhance its functionality, or simply keep pace with changes across API technologies and best practices.



Source: Gartner 2017

The burgeoning Low-Code/No-Code space has become an extraordinarily disruptive force in the domain of API integration - causing confusion amongst developers, users and even vendors. But even the terminology itself can be misleading, as the distinction isn't about whether people need to code or not - but rather about the types of people using technology and tools to build API integration

Representing the "No-Code" domain are the "citizen integrators"—an analyst term to mean business users who can build functional, but perhaps simplistic workflows without having to write a line of code.

In the "Low-Code" domain, we focus on developers instead, demanding platforms to help streamline and simplify their work—delivering enterprise-grade API integration without having to start from scratch.

Twenty years ago, application integration was implemented by hundreds of consultants and experts at a multi-million dollar cost. Now, with no code/low code, the same integration use cases can be implemented at a fraction of the price and in a fraction of the time—and users expect it.

API providers must start to deliver more tooling that expands their addressable market, makes it easier to get started quickly and transfers the burden away from the application buyer.

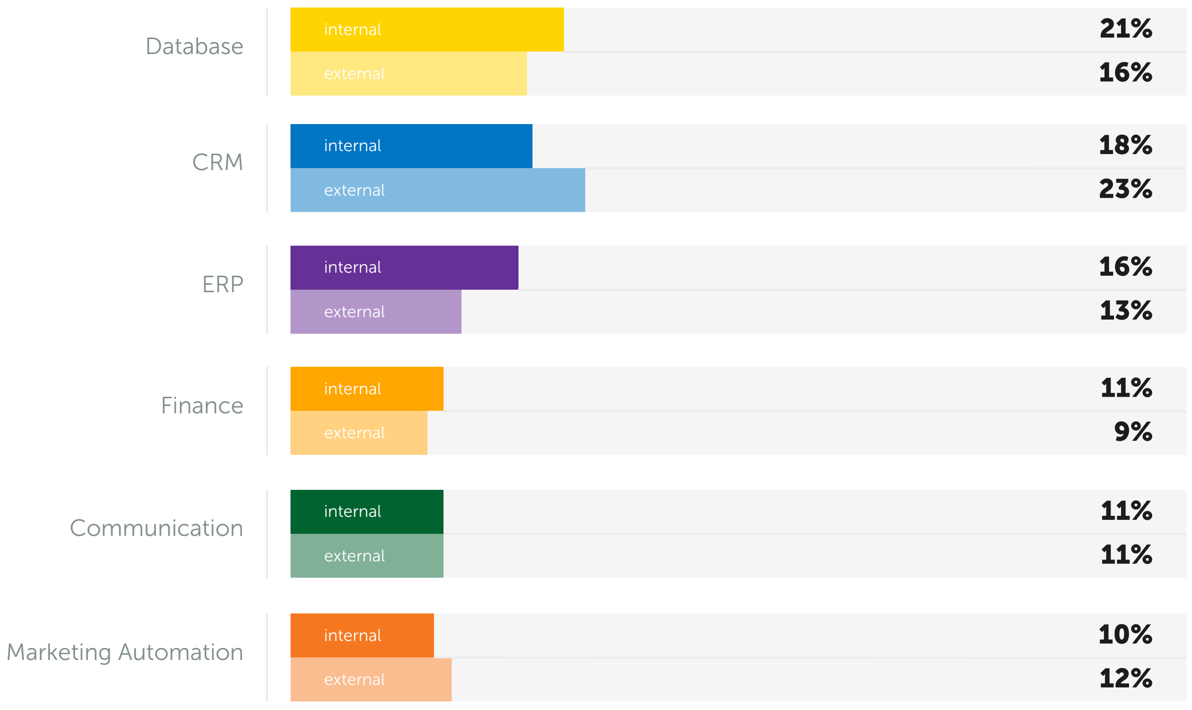
Strategic Shift of Integration

Integration is all about making different apps and systems work together. Allowing them to share data, orchestrating business workflows, and coordinating how users work across growing number of apps.

As the app ecosystem developed, it spawned a new world of API-based integration and a landscape of integration platforms aimed at helping developers manage how their applications interact. Now, as API integration matures, the requirements, technology and integration use cases have also matured. Many organizations must now work to integrate both their new SaaS applications and legacy systems in the same developer friendly environment.

Integration Needs Comparison

Internal Needs within an Organization vs.
External Application Needs by End Customers



SaaS has dramatically changed how we purchase, consume and adopt software—making business apps easy to buy, use, and maintain. As a result, application integration has become a differentiating factor of every business, not just a tool. Today, an average enterprise uses 1,181 cloud apps across its entire organization (source: **Netskope**, Feb 2018). This is a stark contrast to the 1990s, where an average enterprise used 5-10 enterprise apps.

API integration is centered around adopting best-of-breed apps and systems by enabling smart, fast, adaptable integrations and automations across all of a company's apps, APIs, data, people and devices to deliver innovative, personalized experiences.

For API integration that means the way businesses work has drastically changed from the days when traditional integration tools were created. Business apps have become consumerized, self-service, and low-cost, but the tools to integrate these apps often remain complex, technical, and sometimes more expensive than the apps they are integrating.

Integration tools should be flexible, allowing integrations to be changed at any time. In the past, integrations were 'built-to-last'. Once you have your Order-to-cash process going between your SAP and Salesforce, you don't go back to change it—the requirements were more stable. This is no longer the case. Business apps and the processes and workflows around them are constantly changing as new apps are added and existing apps are changed. Constant change is the norm, requiring organizations to think strategically—not tactically—about API integration.

In short: traditional integration platforms may be powerful, but they are limited by the apps they support, the number of use cases they support, and who can use these platforms. These limits are the reflected on the entire organization, making it impossible to enable enterprise users to build integrations that support their line of business, to offer embedded integration experiences their customers need, or to build the products their organization needs to survive in the digital era.

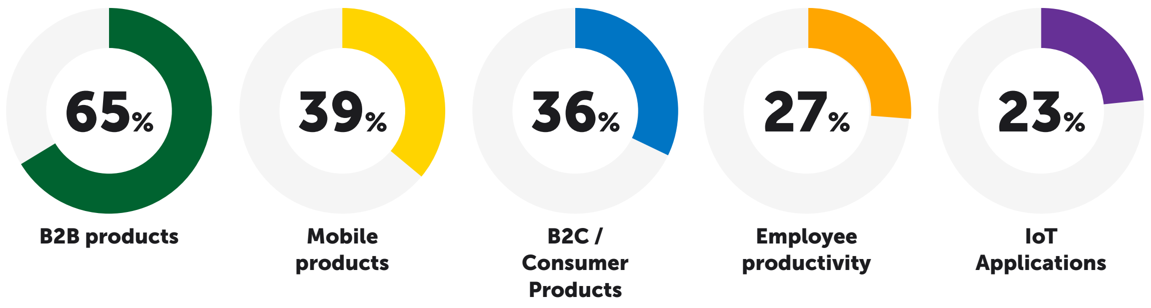
SaaS has dramatically changed how we ***purchase, consume and adapt software***—making business apps easy to buy, use, and maintain.

Evolution of B2B Integration

Businesses are not islands. Connectivity, collaboration, and success with partners and customers is the lifeblood or fuel that drives any organization. Providing a way for these partners to access your products, and exchange information is fundamental. However, many organizations still rely on outdated and outclassed B2B infrastructure to exchange this crucial information. Partner integration in 2018 and beyond requires a different approach. Web APIs not only deliver more flexibility and speed, they will also enable you to attract and connect many more partners and customers than today. During 2018, more than 50% of all B2B collaboration will take place via API integration, and prospective partners will expect that they can integrate with your business via a lightweight API-based approach.

This trend is reflected in our survey results, showing that 65% of respondents are developing new B2B products and B2B API integration solutions.

New Digital Products Being Developed in 2018



While traditional B2B integration via EDI and other similar interfaces is expected to remain a large part of the partner collaboration market for the next few years, API integration technologies are seeing growing adoption across many industries. By migrating to API-centric partner integration, or adding this option to existing business processes, organizations can deliver the same quality of service with much less overhead.

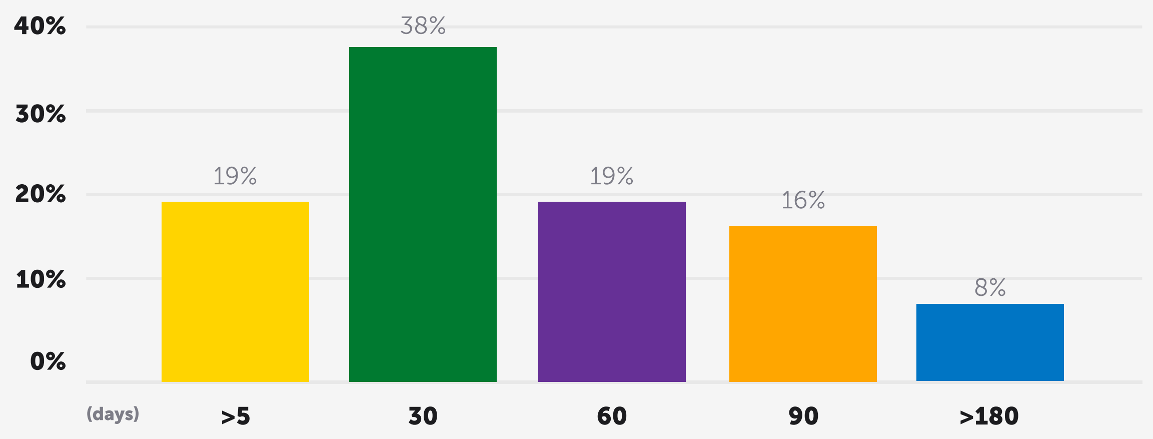
APIs offer an easy way for companies to establish these essential integrations with customers and other businesses. There are 2 key areas B2B integration strategy should consider:

Flexibility

By allowing a multitude of companies to leverage your platform for a variety of use cases, your company can open the door to a larger market segment. Think about how your customers or product teams consume your API. What level of burden are you forcing upon your customers? Would better documentation allow people to integrate with your business better?

One of the biggest mistakes you can make is assuming that your API offers all of the utility that every persona might require. In fact, our survey shows that many developers take weeks to complete an integration—and every day spent is lost business value.

Average Number of Days to Build Net New API Integrations



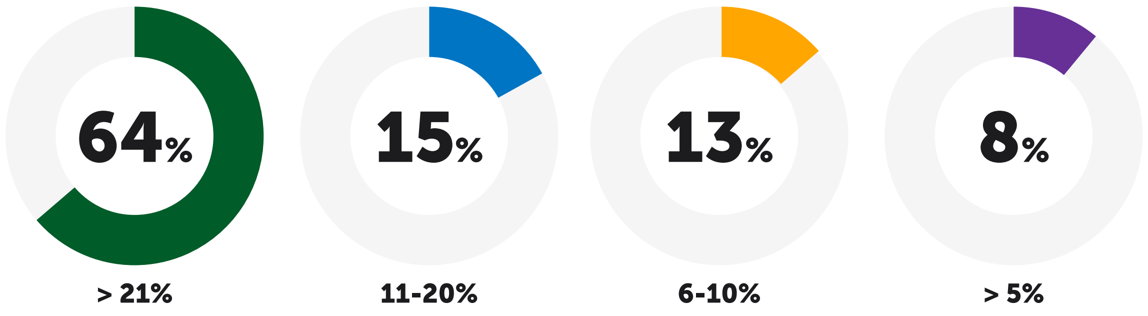
Longevity

According to Constellation Research, 52% of the Fortune 500 have merged, been acquired, gone bankrupt, or simply fallen off the list since 2000. In order to survive in the API economy, you have to think about longevity.

To set yourself up for success, it's important that your organization can support a wide variety of use cases in a dynamic market. The narrower your supported use case, the less likely your company will last in an ever-changing environment. As use cases and needs change it's important that your company has a wide enough lens to adapt in real-time. The best way to widen that lens is learning from the ecosystems and the partners that you do business with. By quickly integrating into multiple application ecosystems you diversify how similar use cases are used in different environments, increasing your stickiness and preserving your longevity in the market.

Offering Integrations Leads to More Revenue

Majority **(64%)** agree that at least a quarter of their user base will upgrade or renew, if they offer integrations



SECTION 2

Summary

API integration is no longer just for developers. Businesses can leverage platforms to drive various competitive advantages and achieve long-term success.

It is important to consider how your organization interacts with all segments of your customer base, delivers and consumes value in the surrounding ecosystem, and thinks strategically about integration.

SECTION 3

Developer API Integration

SECTION 3

Developer API Integration

Today's software architecture draws together a complex web of internal and external services to deliver new value to customers at a lightning pace. Any successful, forward-thinking business today leverages the power of APIs to integrate a wide range of services, data, external providers and internal business capabilities into new workflows, mobile applications, and customer-facing products.

However, as application and end-user demands evolve, this architectural approach also comes with challenges. In this section we'll discuss 3 key architectural concerns you should consider when building, integrating or managing your APIs.

Event-Driven Integration

Based on our research, one sentiment is clear—many architects and developers have high interest in and adoption of event-driven integration.

Most organizations aim to create their own "Composable Enterprise" and will rely on multiple cloud-based services to deliver this vision. Yet, many of these SaaS applications are islands that haven't been optimized for integration, and many developers depend on polling–lots and lots of polling–to create the workflows and integrated scenarios they need. For many application providers, there is a long way to go before they can offer the features and event-driven capabilities developers expect.

Real-time demands are driving event-driven requirements

The need for—and increased interest in—event-driven architecture is arguably attached to the demand for real-time application integration and data movement. The world of "real-time" is perhaps misunderstood, used in the wrong context or at best an overloaded term where the tolerances for delay or latency vary based on the use case.

There can be varying opinions on the definition but in this case, 'real-time' implies web technologies that enable applications to react to relevant business events as they happen—or at least within an understood period of latency. (e.g. 4ms delay may render a financial trading platform uncompetitive, while 400ms delay is easily tolerable for most application integration use cases.)

Event-driven technology

The good news is there are a variety of technology options and architectural best practices available to embrace real-time integration. The WebSockets standard is now well established as an efficient mechanism to stream data in real-time, and Webhooks is gaining momentum as the protocol and pattern of choice for event-driven APIs. The arrival of HTTP 2.0 and Server-Sent Events also represent an ever evolving landscape of technologies that may be used in other event-driven use cases.

Server-Sent Events

First-off, the term **Server-Sent Events** (SSEs) may be unfamiliar for many. For the past few years the specification has been in a state of flux and overshadowed by the more fully featured communication protocol - WebSockets. However, the idea behind SSEs should be familiar: allowing a Web Application to "subscribe" to data updates originated (sent) by a server, and subsequently be notified as these data events occur. The problem with SSEs is that, while they remove the polling technically, they also take much of the control from the user and forces the user into a passive state.

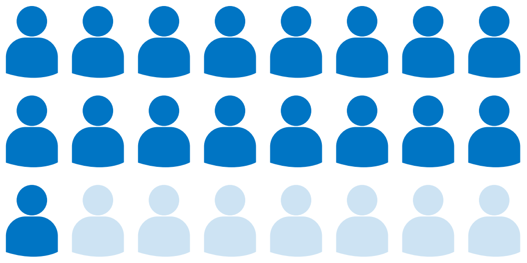
WebSockets

WebSockets provide a richer protocol to perform bidirectional, full-duplex communication. This two-way channel is more attractive for things like games, messaging apps, interactive experiences, and for cases where you need real-time updates in both directions. But again, the solution isn't perfect as the integration pattern effectively breaks the semantics of HTTP.

Over the past 18-24 months there has been increased chatter about the limitations of Web APIs, and how architectural choices must be made on merit versus conceptual elegance. This has led to the evolution of standards—such as v3.0 of Open API Specification, which now includes a way to document Webhooks as part of a RESTful API.

Webhooks

Webhooks work by automatically posting new event data to a user-defined URL, which is monitored by the user's linked applications. Each time a new event is posted to the URL, the linked applications update to include the new data. Unlike polling, which delivers usable data on less than two percent of requests, webhooks only update when new information is available. Because of the increased efficiency, over 82 percent of developers surveyed by Wufoo indicated that they would rather receive new data via Webhooks instead of with polling. Despite this preference, our research revealed only 29 percent of APIs currently support webhooks.



Developers Prefer Webhooks Over Polling

Over **82%** of developers surveyed by Wufoo indicated that they would rather receive new data via Webhooks instead of with polling. Despite this preference, our research revealed only **29%** of APIs currently support webhooks.

Our Recommendation

There is no single design decision that perfectly answers every integration question, but the perils of polling are clear: higher network overhead, increased cost and poor user experience. Our recommendation is to consider alternatives like Webhooks when building your next Web API or integration scenario.

An Event-Driven API Landscape Driving What Matters To Consumers

Contributed Content from Kin Lane, API Evangelist

One clear signal emitted by the strongest API providers across landscape in 2018 is the importance of investing in event-driven infrastructure, and the development of a real time perspective of getting business done using their API-driven infrastructure. The phrase event-driven will mean different things to different providers, but generally it reflects the ability to respond to internal, partner and consumer needs in real-time, as events around valuable resources occur.

Event-driven infrastructure might involve the development of Apache Kafka pipelines within the enterprise, the delivery of financial market data to web and mobile devices using Server-Sent Events (SSE), all the way to pushing data and content via webhooks as resources are added, updated, or even removed via APIs. API infrastructure has historically been centered around making the resources they have available in the web, mobile, and device-based applications. Event-driven API infrastructure is about making sure the most meaningful, valuable, and relevant information is delivered when and where they are needed--driven by API consumer's needs and desires, moving the conversation beyond just what resources API providers have available, and making it about what consumers need.

When you watch what the most seasoned, and mature API providers are investing in across the landscape in 2018, you hear stories about Yelp or Pinterest's investment in Kafka infrastructure internally. You also see Stripe, and Slack adding further to the list of webhook events that their API consumers can subscribe to. Event-driven signals are a sign of platform maturity, and that API providers are moving beyond the early stages of their development, having seen the traction and adoption that they were seeking. These providers are now interested in cutting through the noise and distortion, and ensuring that their API consumers are subscribing to and in tune with the most relevant events that are occurring via the platform at any moment.

Event-driven architecture isn't just about data firehoses and pipelines, or real-time streams. It is about making sure your platform is pushing beyond the basics of API operations, and is managing and moving around data, while applying algorithms in the most efficient and meaningful way possible. Realizing an event-driven way of doing APIs isn't about the next trend in doing APIs, it means you are mastering the regular aspects of platform operations, and have begun investing in the next stage of its execution, speeding up the delivery of resources, and optimizing their orchestration. Event-driven APIs aren't just about being able to technically deliver in real time, it is about being able to do business in real time, responding to what matters, as it occurs.

Emerging API Styles

As the API integration landscape evolves, new API styles such as OData and GraphQL are emerging and gaining ground amongst developers. Here we'll evaluate each of these with regards to achieving interoperability across APIs for analytics, integration and data management.

What is REST?

REST (representational state transfer) or RESTful web services are one way of providing interoperability between computer systems on the internet. REST-compliant web services allow requesting systems to access and manipulate textual representations of web resources using a uniform and predefined set of stateless operations. RESTful implementations make use of standards such as HTTP, URI, JSON and XML.

It's important to note that REST is an architectural style, not a standard.

What is OData?

Originally developed by Microsoft in 2007, OData is an OASIS standard REST API well-established among companies such as Microsoft, SAP, IBM and Salesforce. It allows the creation and consumption of queryable and interoperable RESTful APIs in a simple and standard way. OData gives you a rich set of querying capabilities and is quickly gaining ground for its open source approach, as well as its exceptional scalability.

What is GraphQL?

Developed internally at Facebook in 2012, before public release in 2015, GraphQL is a data query language deployed at companies such as Facebook, Shopify and Intuit. GraphQL provides a complete and understandable description of the data in your API, gives clients the power to ask for exactly what they need, makes it easier to evolve APIs over time and enables powerful developer tools.

Query Capabilities: OData vs GraphQL

Operations	OData	GraphQL
Filtering	✓	✗
Ordering	✓	✗
Aggregation	✓	✗
Joining	✓	✓
Paging	✓	✓

The table above highlights common criteria for accessing data through open standard interfaces. OData has the full range of support for all these query capabilities. You can do some of these operations with GraphQL, but they're not standardized or documented in a way to achieve interoperability.

GraphQL is much like REST in that it defines the way to interact with a web service, but it doesn't tell you what the service does. So you can do filtering, ordering, and joins, etc, but the application developer has to understand how those work semantically to be able to know what their behaviors are.

API Versioning and Maintenance

One key area of maintenance is handling updates to applications when the API changes, while maintaining older versions. The biggest problem leading to this dilemma with REST APIs is that all of the fields are returned when you query an endpoint. API providers have no visibility into whether or not clients are relying on information in specific fields. API consumers must process all of the fields returned even if they do not need the information.

GraphQL has addressed the problem of API versioning and maintenance by forcing clients to specify exactly which fields they require.

OData provides similar functionality by providing a select list to limit the number of fields returned to those needed by an application. This reduces response size and processing in an application. However, it does not provide a mechanism to indicate that fields are deprecated.

GraphQL: Changing the Landscape of APIs

Contributed Content by Mark Boyd

GraphQL has already seen rapid adoption since its public release in 2015. Production use cases from enterprise, government and startups are already in place and new tooling continues to evolve and diversify. As an API technology, GraphQL is finding its place in the broader API market landscape and has potential to strengthen adoption by leveraging two other API sectors: in Serverless and in the Internet of Things.

While sectors like ecommerce and publishing are already seeing significant value from moving to a GraphQL API architecture, this year is starting to see other industries take notice, particularly online video services. Key early user segments in marketing technology, software development tooling, and digital consultancies will help spread adoption to other industries.

One of the key benefits of GraphQL—the ability to create a data abstraction layer that can combine multiple sources through a single gateway and endpoint—will garner growing interest from industry sectors that have complex data supply chains. Data science, healthcare, and city services will be drawn to GraphQL in 2018 in the same way that publishing, social media and ecommerce have been to date.

GraphQL has quickly established itself as a valid option for businesses and developers making choices around how to create and manage their API strategy. Across the sector, the biggest gap is a matrix decision tool that allows API creators to assess each type of API architecture option and weigh up the best cases each is suited to implementing. Without this clarity, each individual business must do their own research from REST, Hypermedia and HATEOAS, SOAP and GraphQL options.

For many use cases, GraphQL has the potential of leapfrogging the growing interest in hypermedia APIs. In hypermedia APIs, responses describe what data and links are available so that as queries are made, an application can discover what other data and capabilities are available that are relevant to the application's need.

In hypermedia, this requires good metadata and well structured endpoint linking, but can return bloated responses that slow down an application's performance. In GraphQL, the schema itself describes the full data model. The introspection quality means that inside any GraphQL API is the ability to query its own data model to better understand what data is available. And GraphQL's advantage of allowing clients to only call the specific data that is needed avoids hypermedia's bloat problem.

As an abstraction layer that is able to draw in multiple data sources and make them available or referenced within a single schema/data model, GraphQL has huge potential for use in sectors where data comes from disparate and disconnected sources. GraphQL removes data access barriers and makes all data—regardless of the original owner or place of publication—available through a single schema where relevant, related data can be queried at the one time. This suggests that the fields of data science, healthcare, city services and other sectors that manage complex relational data models may see advantage in adopting GraphQL.

There is no doubt there are still some challenges GraphQL must solve. More complicated operations such as writing data in bulk, managing real-time data querying via a subscription rather than polling mechanism, and managing caching and pagination are partly solved but require more testing against production use case needs. Authorization stands out as one area of work that requires significant additional tooling and a variety of options.

Ultimately in 2018, GraphQL's evolution is still in its infancy, but with the rapid interest from developers and the quick iteration and availability of tooling, there are strong signs that GraphQL is becoming a legitimate option for how APIs are created and managed, offering significant performance and developer experience benefits.

API Security

APIs are certified "cool" - they provide access to data and services otherwise locked behind corporate firewall, they enable easy development of web & mobile products, and they allow organizations to do business in new ways. But this strategic opportunity opens up vulnerabilities, and APIs are becoming the primary attack vector for unscrupulous individuals, organizations and even governments!

Security then, must be a leading concern. All API stakeholders (developers, architects, operations and the business) must play a role in deploying a security model that both enables and protects. This is a list of guiding principles that API projects should follow:

1.
Security features, like deciding how to deal with authentication and access control is often relegated to the last design consideration. But a robust security model must also consider your overall application architecture and find all layers where threats and vulnerabilities may exist. It's also important to think about reusability in your security architecture, making it easy for developers to leverage standardized mechanisms at each application tier.

2.
Basic access control is actually pretty simple. Users and clients request to access Objects like data, applications and services. Access controls then make a decision to grant or deny that request. Today, user management is a mature space and most organizations have a central user directory. Object management must reach this same maturity, enabling better controls and improved granularity across your APIs and data model. Leverage existing standards (E.g. OAuth 2.0) to grant granular access to services.

3.
Many believe security architectures are focused on keeping the bad guy out, but in fact most often the inverse is true. Access control infrastructure is designed to provide a user experience to authorized entities: employees, customers, etc. It's important to design for malice: assume that those trying to gain unauthorized access are in fact not "playing by the rules", and instead finding ways to work around the areas governed by policy.

4.

Users simply don't care about or understand security—they just want easy access and peace of mind that their data is secure. Unfortunately, for a long time security measures and technologies have delegated far too much responsibility to the end user, expecting them to make educated and technical decisions. How often have you been asked "Do you trust this certificate?" in a web browser? It's not practical to assume users have the technical knowledge to understand PKI—so how is it reasonable to push this responsibility to them? API security should be designed in such a way to balance user experience and security.

5.

Developer experience is just as important as user experience, and knowledge around security can be dangerously lacking here too. API consumers can't be expected to know security, meaning your APIs can unwittingly create vulnerabilities by not arming developers with sufficient knowledge. For example, a developer may hit an API too many times and degrade performance. Or a developer may not know why or how to protect API and session keys. These can result in Denial of Service, and other security issues.

6.

Finally, API integration patterns are evolving, and while request/response still makes up the lion's share of implementations today, there are other models that need to be secured as well. With an increasing preference for Webhooks, Server-Sent Events and WebSockets, the client-server designation is becoming blurred. Signatures and encryption (E.g. TLS) should be considered to help protect these bi-directional integration patterns.

DevSecOps Approach to API Security

Contributed Content from Isabelle Mauny, 42Crunch

APIs are doorways that have enabled organizations to build new connections into their monolithic software and data systems. Of course, doorways work in both directions. So while an enterprise may expose its data assets and business capabilities using APIs, they are also making extensive use of external services in their own systems by relying on APIs from third parties.

This multiplication of endpoints has blurred the security perimeter of the organization. The threat surface has increased and more security risks need to be managed. A manual, ad-hoc approach to securing the numerous and evolving endpoints simply does not work. Furthermore, when API security is tedious and not agile, it is considered as a roadblock to quickly delivering apps and innovations and therefore often left out of development and operational discussions.

To address both issues, security teams must deliver “security as code”, that is, translate security requirements into code that can run automatically at every step of the API lifecycle. This can be done by taking a DevSecOps approach and “pushing security left”, considering security aspects already when doing the initial design and defining requirements setting phases.

The DevSecOps mindset considers security issues as bug fixes. Security vulnerabilities are essentially bugs that can lead to unwanted outcomes in an application or a product. Like all bugs, you want to discover them as early as possible. The overall cost of fixing a bug is up to 30 times higher if discovered at production time rather than at development time.

At design time, organizations must take these core actions to protect the APIs:

- Development, security and operations teams work together throughout the API design cycle, discussing and building the API contract (preferably based on open standards like OpenAPI), deciding where the API will be deployed, and who will use it. Together, they build the API's threat model that will drive the security policies to be applied to the API.

- As APIs are designed, a set of pre-approved, pre-tested security policies are put in place that match the security level required. APIs are tagged depending on the risk evaluated as part the API threat modelling. The policies must address the core security aspects:
    - » Enforcing the proper TLS settings to protect the API channels
    - » Validating the API contract and the flowing data
    - » Configuring the OAuth protocol and OpenID Connect authentication flows
    - » Auditing
    - » Non-repudiation
    - » Encrypting and signing the API payload (optional)

To secure APIs during runtime, the following security aspects must be automatically enforced throughout the CI/CD pipeline:

- Security policies are incorporated as early as possible into the continuous delivery and testing processes, and an API security gateway is deployed to enforce those policies.

- Code analysis tools (static and dynamic) are deployed and hooked to the CI/CD pipeline.

- APIs are actively scanned APIs for vulnerabilities, in other words: **you must hack yourself!** You can start with **ZAP**, an OWASP open source penetration testing tool that you can plug to the CI/CD pipeline.

- Deployments are tested for proper TLS configuration using tools such as **SSL Labs** or **securityheaders.io**.

- Software and libraries are constantly kept up to date, and tools are put in place to actively monitor security vulnerabilities.

- Systems are constantly monitored for security alerts in order to respond promptly to attacks.

In addition, it is critical to educate personnel on security risks: all concerned parties should at least know about the **top 10 OWASP risks** list and how to address them.

Protecting and testing your APIs through a repeatable, well-defined, automated process enables the enterprise to continue to innovate at the pace required to stay competitive, without compromising on security.

Interactive Calculator

Score the Current State of **YOUR** API Integration

To wrap things up, we wanted to give our readers an interactive calculator to gauge how your API currently stacks up against the current state of API integration. Gauge how easy it is for API consumers to interact with your service, and how easy it is for data to move between your service and other applications. Where do you stack up against the rest of the API community? Share your score and be sure to mention **#StateofAPIIntegration**.

CLICK HERE TO SCORE YOUR API
https://hubs.ly/H0bq4kv0

Closing

And there you have it. We hope you enjoyed seeing the data and hearing from our contributors about the trends that are set to affect APIs and application integration. Follow the conversation by using the hashtag #StateofAPIIntegration.

Check out our resource center for other easy to follow information. If you have questions, or would like to learn more about Cloud Elements, contact us today!

CONTACT CLOUD ELEMENTS