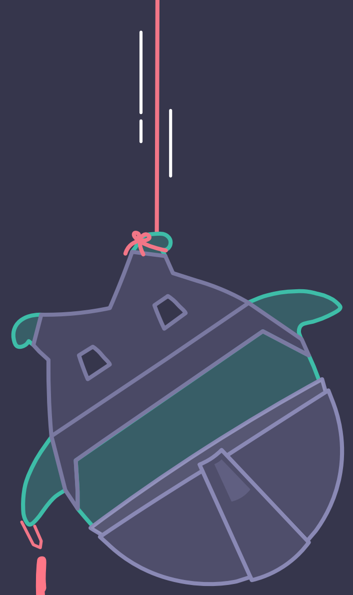


THE Opinionated GUIDE TO API MANAGEMENT IN 2023

Tyk

Contents

00 Introduction

At the Tyk API Extravaganza in December 2022, Tyk Product Evangelist Budha Bhattacharya grilled Tyk CEO Martin Buhr and Executive API Consultant James Higginbotham on what's hot – and what's not – on the API scene and in the wider tech world right now.

Add into the mix a plethora of additional super-brains – Nathaniel Okenwa, Developer Evangelist at Twilio, Claire Barrett, Director of APIsfirst and Co-founder of Women in APIs, and Tyk COO, James Hirst – and we've got ourselves a pretty extraordinary roster of expert minds to dissect and pinpoint the areas to focus on for the coming year, and beyond.

Here, we dig deep into the API hot topics for 2023. Everything from microservices, GraphQL and service mesh to how to create more diverse, inclusive and equitable organisations – and why it would be nice to have a very boring 2023 - so get strapped in, folks!

With opinions from

01 The muddy waters of service mesh v/s API gateways

There needs to be more clarity about the differences between a service mesh and an API gateway and which is best. Service mesh was designed to help deal with the microservice pattern (because the microservice pattern is tough to manage).

Service mesh is a meta-network that sits on the existing network. It's a sort of increasing abstraction of complexity – you have to learn Kubernetes and understand the microservices pattern. Once you've deployed that, you must implement the service mesh and understand that. ‖ Feel the heat, watch the video ▶

Not something for the faint of heart. Overall, though, service mesh isn't a bad thing, according to Martin. It's a technology designed to solve the problem of popular software architecture: microservices—more on those in a moment.

Meanwhile, API gateways are specialised proxies that are laser-focused on doing API work and solving problems in that space.

01 An extension of your API gateway

A service mesh is an extension of an API gateway in that it is a proxy that is laser-focused on connecting multiple systems. This makes system calls possible, creating observability and security layers across thousands and thousands of microservices.

A service mesh and an API gateway do very similar things at their core. They move traffic from A to B in a certain way, creating observability and security. The critical difference is that they try to solve two very different problems. The gateway is focused on securing your API surface area from the outside world (north-south traffic), while the service mesh is very much about east-west traffic.

And, of course, it's not an either-or situation. Tyk, for example, works beautifully with a service mesh

Service meshes come in different forms, from very minimal to more complex. You need to ask yourself what you need from a service mesh. What are you trying to achieve? Thinking differently about how you architect your software could lead to less complexity and less management overhead. ‖ Feel the heat, watch the video ▶

02

Microservices - is it a force for good or evil?

Are microservices overblown? Yes, they can be a force for good when used in the proper context. Facebook, Netflix, Amazon and the like probably really need them. But do small and medium-sized organisations? Do start-ups? Unlikely.

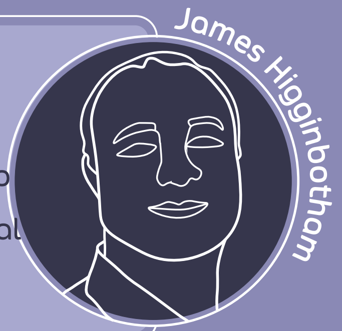
It's an example of group-think and trends dictating what the market should think is good.. A microservices approach is sometimes cheaper or better; in fact, it can be more expensive to maintain. Using microservices when you don't need them is a problem. || Feel the heat, watch the video ▶

Again, this comes back to thinking differently. To avoid falling into the trap of adopting technology to follow a trend. Many API programs and digital transformations have been driven by a perceived need to move from monolith to microservices over the past couple of years. But is there a better way to do things?

02 Design software and be thoughtful about it

Various approaches can work. You could have multiple monoliths within your organisation and smaller or mid-sized monoliths that can be managed appropriately and modularly designed.

What we've lost in the software engineering world, with the advent of all these frameworks and nice little CV builders and everything else, is the ability to design software and be thoughtful about it. We lost the ability to remember loose coupling, high cohesion, and some of the core foundational principles of software design. ‖ Feel the heat, watch the video ▶



Indeed, there is a time and a place for choosing microservices. But the reasoning behind doing so needs to be sound. Organisations need to understand the pros and cons, the architectural principles behind choosing microservices and what trade-offs they are making when adopting them.

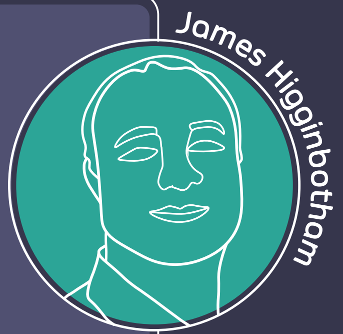
03

GraphQL - is it going through an existensial crisis?

We're at an exciting point in the evolution of GraphQL. Many who have adopted it are using it because they need the speed of development between the front and back end – they want something very efficient and effective to handle both sides. Plus, there's the interesting GraphQL federation use case that's starting to emerge. Certainly, GraphQL is solving problems for some people.

However, it's not all smooth sailing,

> From the operational side of it, there are a lot of challenges. And I see many people who are short-sighted – they don't realise what's involved in rate limiting it, securing it with fine grain authorisation, data entitlements and so on. I think it's in this maturity phase where some people use it successfully, and others are hitting walls. || Feel the heat, watch the video ▶



03 GraphQL - is it going through an existensial crisis?

For those old enough to recall, there are echoes of the SOAP challenges from 20 years ago. So, will GraphQL evolve, or will we have to keep shoving more middleware solutions in there?

GraphQL was initially designed to solve the one-to-many problem of single-page web apps and mobile applications.

It's very reminiscent of the SOAP days and RPC, where you make a single query and get lots of data back, but you don't know what's happening in the background. Now, you've got microservices showing up and need to scale GraphQL across microservices. Hence, the solution becomes GraphQL federation, an additional layer of complexity on top of something already quite complex. || Feel the heat, watch the video ▶

03 The financial considerations

There are financial considerations, too. GraphQL is expensive to implement on the server side, mainly if you're coming from REST. Yet increasingly, it is moving up the stack into the server side in terms of its use case, facilitating calls between systems that are GraphQL based instead of client/server.

Tyk has developed an excellent GraphQL tool as part of this future vision for GraphQL called Universal Data Graph.

Universal Data Graph enables GraphQL in the server without the need to write a bunch of resolvers and deal with all of that complexity – you plug your existing infrastructure into it and immediately take advantage of GraphQL. GraphQL is designed to connect all of your microservices into a single query. This provides an inspiration pattern for integrating multiple services for your developers to query everything – to query almost all your microservices as a single database.

You can, of course, try to do the same with older patterns. Try with an enterprise service bus or SOA (for example) or just by writing lots of proxies or transactions where you group REST calls into batches and try to get them to work or write complex middleware; instead of doing that, you can simplify it with GraphQL. Your front end can use it, and your server-side can use it, and all of a sudden, you have this magic mirror to all your data sources. It's very cool, but there is a GraphQL learning curve. || Feel the heat, watch the video ▶

03 The financial considerations

Of course, most people look at GraphQL alongside REST services. That means additional complexity, as they want to use GraphQL to develop but not expose it to customers due to worries about security, rate limits and cascading failures (due to all the different routes involved). So then you end up building REST resolvers to call into GraphQL (which then calls into REST), meaning you end up with a bizarre level of complexity.

What does the future hold for GraphQL, then? We need to get GraphQL right and secure it so anybody can consume it on the client side. Or, accept that it's an excellent server-side component and that there are solutions to scaling – such as federation – that are very elegant once you adopt them—provided you solve the client-side access.

The key takeaway? Think carefully about use cases and about decomplexifying your solution. Is GraphQL the right tool for the job?

04

You can't trust technologists to build APIs alone

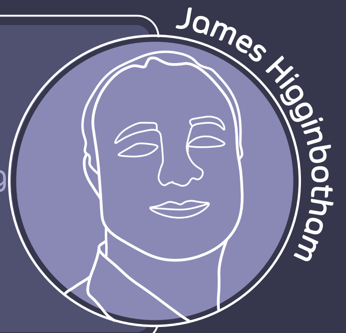
The other theme that is top of mind is whether developers/technologists should be left alone to build APIs or whether there is value in adding other voices.

What we design as an API impacts others. If you're a developer working inside a single code base and agree with your team to use Java, Node.js, Ruby, Python, Golang or whatever, the risk is isolated to the team working on it.

However, when you design an API, you open that up to other teams in your organisation that will use the API or to partners or customers. That means you have communication paths, and you have dependencies that are outside your control.

As such, API design sits at the intersection of business, product and technology. It means you have to think about more than just the tech – more than just getting excited about the specific service you're building. You need to consider the contract you're building with the user, thinking about what the API delivers and how you can communicate how to use it.

We need to ensure we take advantage of the opportunity to solve real problems for real humans by focusing solely on the technical elements of the design. API design has to be cross-functional, with people representing the business and the product involved. Don't immediately start coding and see what your API design evolves into. ‖ Feel the heat, watch the video ▶



04 The balance between innovation and standardisation

The balance between innovation and standardisation plays into this. Indeed, it's something that every company faces, not just about APIs. There's always a choice between designing fast and breaking things (and accepting that in doing so, you're embedding technical debt into your future growth) or taking the time to design your solution correctly.

Much of this depends on your business model. If your model is just about the API and the developer experience, you need a design-first approach, as your API is your product. You could go fast and break stuff for something you're building for services internally that supports a more comprehensive, external-facing product.

It all depends on who your consumers are. Just be sure to set expectations if you're going to go fast and break things, so people know what could happen!

If the API is your product or a significant revenue component, design it well! Think about who's going to use it, how they're going to use it and how they're going to interact with your organisation programmatically. ‖ Feel the heat, watch the video ▶

Of course, these are two extremes. Most businesses take an approach closer to the middle, balancing the length of product iteration cycles with getting to market before missing the boat.

05 The wider tech scene

Stepping out of the API management world, there are other central themes of the moment. Chat GPT and OpenAPI seems to be on everyone's radar right now, as we've entered a new phase of accessibility for this type of technology for the average consumer (instead of the more techie developer type).

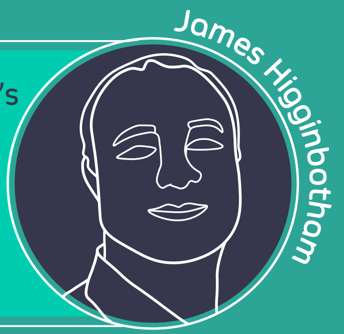
These synthesised models that can create things out of the corpus of human knowledge and language are fascinating. We're on the cusp of a rapid evolution in tooling. For example, there's already a tool to comment on your GitHub commits with Chat GPT. Tools will slowly but surely creep down the value chain – it's already happened in gaming with AI upscaling processes built into graphics cards.

I think the biggest sea change 2022 brought in terms of tech is this magic black box that is AI. We'll see some interesting tooling built on top of it that will be very cool but also exceptionally creepy! In terms of coding and developers, AI means that we may need to change how we think as new possibilities come to a head. || Feel the heat, watch the video ▶

05 The wider tech scene

There were lessons to be learned from the whole Twitter situation. One was that many of us are looking for the lowest friction way of getting things done. The other was that people gravitate towards centralisation.

We have this tendency to find where everybody's at or where everybody's going and want to go to a central authority to try to find something. These lessons could apply to us in the API space, API management space and start-up space. ‖ Feel the heat, watch the video ▶



Of course, the issue of centralisation is also top of mind due to Web 3.0. Web3 brings with it a major mind shift around decentralisation. How do we merge that with what people's psychology will allow them to do at this point in time?

06 Remote working and its distinct advantage

Remote-first companies have no office, so there is no ‘norm’ that people must comply with consciously or otherwise. Instead, everyone works in their context, whoever they are, and wherever they are – there’s no culture to have to ‘fit in’ to. It becomes a question of looking for common ground that unites people, despite very diverse values and experiences.

Many still see remote and flexible working as a perk rather than an improved operating model - invested in the tired idea that it is an employee benefit that companies can no longer afford.

This is based on the assumption that remote and flexible working means ‘less working’. And a belief that is forcing folks to travel through rush hour, to sit in an office and be monitored by a ‘manager’, before crawling home in traffic again, will be the best way of dealing with a difficult economy and increased pressure on productivity.

Remote and flexible working is not a ‘perk’; it’s a better way of working, a win-win for the team and the employer. In a worsening economy, the managers who can only show impact by dragging folks into an office are the overhead companies can no longer afford.

07 Diversity, equality and inclusion

Diversity, equality and inclusion in the tech industry is a topic very close to our hearts here at Tyk.

Studies show that teams with a more equitable gender balance are generally more creative, resilient to change, financially successful, and have a higher overall engagement. Claire Barrett summarises why DEI is good for business and what it looks and feels like when it's going well:

From an engineering and product point of view, ensuring diversity in teams provides diversity in thinking and collaboration practices that go beyond the boundaries of traditional enterprises. It's the best way to solve problems and create things differently. || Feel the heat, watch the video ▶

07 Diversity, equality and inclusion

Based on Tyk's experience operating a global business with a remote-first model, thoughts on building a diverse, inclusive organisation are front and centre. Tyk has staff in 26 different countries, based on every continent except for Antarctica. It also has a 36 percent female workforce. There's always room to improve, but the organisation is "already a melting pot", according to Martin:

> Diversity and inclusion is a big deal. When your team has different backgrounds, ethnicities, genders and orientations, with all those viewpoints represented, it forces people to think about who's on the other side of that chat box or call and how they see the world. That brings a much bigger understanding of the type of users you might be speaking to. If you really want to build a great workplace, you need to ensure that everybody feels safe expressing themselves. And that they are respected. It's not about meeting quotas. Inclusion and treating others fairly and equitably require constant vigilance. ‖ Feel the heat, watch the video ►

Many components come together to make this happen, from establishing the proper hiring patterns to creating an environment that encourages more people to join you. This culture is safe and secure for all.

It's not always easy, and you can't impose cultural values across diverse contexts. There can be a fine line sometimes, but ultimately, it comes down to the organisation respecting differences rather than imposing its views.

08 Engineering efficiency

In 2022, 145,000 tech workers were laid off. With global inflation continuing to be an issue and countries battling recession, it's a rough start to 2023.

However, challenging circumstances also present opportunities. The opportunity for 2023? Efficiency. We are facing a macro-economic environment of cost-savings, efficiency and ROI. This places API awareness on how they affect every aspect of the business and how all teams can use them, front and centre.

To build a high-performing business, we must focus on efficiency and the decisions that tech teams need to make for colleagues and end customers to feel safe and in control. APIM is bigger than code and data; it's about having the tools and processes to achieve higher business and team performance.

08 Engineering efficiency

Here's how you can reimagine your tech workforce for higher productivity:

1. Improve internal engineering efficiency with centralised APIM and reduce management overhead constant vigilance.
2. Reduce time spent by engineers on bespoke API integration solutions
3. Ensure improved reliability, security, uptime, and manageability of digital operations.

09

Nathaniel Okenwa's hot trends

Developer Evangelist extraordinaire at Twilio, Nathaniel Okenwa, spends his days educating and chatting to developers about the APIs they're using, so he's always got one eye on the significant trends and what is coming up next.

He has identified five top trends for 2023. ‖ Feel the heat, watch the video ▶

1

API-first development

API-first companies have an inherent advantage – they can be more productive, integrate faster with partners and grow efficiently. By shaping their strategy, hiring and infrastructure around APIs, companies will be modelling API technology into the very structure of their business on multiple levels.

2

Developer experience is here to stay

If you're not focused on developer experience (DX) yet, you should be! In the world of APIs, the customer is a developer, meaning that the DX is the customer experience. This is leading companies to push for the Holy Grail of self-service, where they can empower developers to build and get to production on their products through self-service.

09 Nathaniel Okenwa's hot trends

3

AI and ML will continue to push boundaries

Artificial intelligence (AI) and machine learning (ML) continue to push boundaries. AI/ML is still very expensive, and we've all become used to promises that have yet to materialise over the past five years. However, we're finally on the cusp of change. Talk of ChatGPT has been going wild on Twitter, and there are some fantastic stories.

4

Demand for API skills

As investments in APIs increase, so will the demand for suitably skilled developers. Many developers are currently working on consuming APIs. Still, as APIs become more critical in our API-first world, developers will need to learn how to architect and build great APIs.

5

A new age of APIM

More and more companies are using API management platforms instead of developing in-house teams. They are looking for efficiency – the ability to spin up APIs at scale quickly but without intensive labour and financial investment.

10 What does 2023 have in store?

Who would have thought that top of our wish list for 2023 is for it to be as boring as possible? But after the traumatic geopolitical events of 2022, we must emphasise global civility and peace. This includes uniting workforces despite very diverse values and experiences so that everyone is seen and heard.

The great thing about the API space in 2023 is that so many organisations are starting to incorporate more of the human perspective into their APIs and thinking about how they will be used, including feedback and input of things beyond the immediate tech sphere.

While tighter budgets will present some challenges over the year ahead, this will also allow some organisations to stand out and make a difference by having the tools and processes to achieve higher business and team performance.

Tell us we're wrong

Speak to our team

Check out the Tyk Community

Call us out on Twitter