

What is API Management?



Robert Broeckelmann · Follow

5 min read · Feb 27, 2016



23



This post was originally published as “What is API Management?” on the Levvel Blog.



In my last post, I explored what APIs are. In this post, I build on that question by answering what composes a holistic API Management program.

What problems is it going to solve? What capabilities does it bring to the table? The initial motivation could come from mobile, improving B2B Integration, SaaS solution integration, or an alternative to SOAP for internal system-to-system communication. I come at this topic with a healthy dose of pragmatism — not particularly interested in the marketing hype, and conscious of the differences we see across enterprise clients implementing these solutions. Too often, these clients struggle to define the scope of API Management, making their definition either too narrow (and missing business opportunities) or too broad (and losing focus on the critical capabilities). These mistakes are costly and avoidable. As with any undertaking in Information Technology, understanding your organization's use cases and requirements is imperative to success.

The scope of API Management includes:

- The process of publishing, promoting, and overseeing APIs in a secure, scalable environment
- Ensuring that developers and partners are productive
- Managing, securing, and mediating your API traffic
- Allowing an organization to grow their API program to meet increasing demands
- Enabling the monetization of APIs
- The intersection of technology, business, organization, and integration concerns

An effective API Management solution manages the use of APIs to open up an organization or a system's data so that it can be utilized by other parts of the organization or possibly third-party organizations in new and useful

ways. A for-profit organization would use these capabilities to either create new revenue streams of revenue or enhance/optimize existing revenue streams (monetizing APIs); a non-profit organization or government organization may be trying to cover its operational costs while fulfilling its mission to provide benefit (in this case, in the form of data/information) to the public. API Management should provide an easy, effective, and efficient experience for the development community that is supposed to be using those APIs. API Management also provides a governance and life-cycle framework for APIs.

From a business perspective, API Management is a revolution; from a technical perspective, it is an evolution of the earlier Service Oriented Architecture (SOA) paradigm. While API management includes much more than the SOA concepts of the early 2000's, those concepts are still present. If anything, API management is partly a response to the flaws in early SOA strategies.

Every API Management solution contains three components:

- API Gateway
- Developer Portal
- Management Portal

Every vendor has these parts — the names may differ. For the larger, more-established companies operating in this space, these pieces were probably built from existing components (for example, WebSphere DataPower acts as the API Gateway for IBM API Management). More on this later.

API Governance is also part of API Management. In fact, it is probably the most important and the least sexy part of API Management — yet, it is likely the most often neglected. Governance is not a bad thing. In fact, for anything to be truly adopted by the enterprise, it must have adequate and functional governance. The key phrase there is “functional governance”. A gauge I have long used to determine whether a new technology, process, or thing is “functional” is whether it solves more problems than it creates. If the answer to that question is no or you’re not sure, rethink the approach.

API Governance addresses:

- tracking the life-cycle of each API from inception to sun-setting (more below)
- tracking the API Consumers and subscriptions (relationships) to APIs utilized
- the API Security Model employed and the details of managing it
- defines the API interface standards used for creating APIs (an organization’s standards for usage of something like Swagger) in the organization
- gathering statistics of both the Developer Portal and API Gateway usage
- utilization-based billing
- API versioning
- JSON (or XML) Schema versioning for input and output data structures
- tracking of routing information

API Governance must tie into:

- Change Management
- Asset Management
- Configuration Management
- Legacy SOA Governance (with the goal of eventually replacing it)

Each looks a little different in every organization. They likely are not managed by the same groups. There will be politics involved. One size does not fit all. Yet, all must be integrated.

API Life-Cycle Management (which tracks the interface's life-cycle, not the implementation) is part of API Governance. The details will vary, but this basically describes the promotion process (life-cycle) of an API version from initial concept, to definition, to the lowest-level development environment, to production, and eventually to sun-setting. As an example, suppose your organization has the following environments:

- Unit Test Environment
- Quality Assurance Test Environment
- Load Test Environment
- Production Environment

Now, let's assume that your API life-cycle captures the following additional steps:

- Inception (identification of a business or technical need)
- Definition (interface definition)
- Development

- Sun-setting

The API interface version's life-cycle tracks these stages together with the deployment to each environment (listed above). For those who lived through SOA (especially with an overbearing SOA Governance paradigm), this should be starting to sound familiar with red flags and alarm bells. This started to become very cumbersome. There is also the question of how this ties into an Agile development methodology?

First, I am making an assumption that an interface-first design (or something in line with Apigee's API First mantra) methodology is being followed. Meaning, you take the time to design the API interface and data structures before a single line of code has been written. This is like the top-down development methodology of SOAP Web Services. Next, I'm assuming that by the time Development and Unit Test deployment stages are hit, the interface is stable (yes, something always slips through the cracks). Next, I am assuming that the API Management Platform provides a self-service feature that allows for relatively painless transition to the next phase of the life-cycle; this deployment should be automated and allow for self-service. Whether this aspect of the API-side development process is managed through the Developer Portal, DevOps management tools, or something else depends on the implementation. There is a sweet spot for every organization regarding how elaborate this should be.

So, that represents my definition of API Management. I must believe there will be passionate opinions both agreeing and disagreeing. Leave comments.

APIs

Api Management

What Is Api Management



Written by Robert Broeckelmann

1.99K Followers · 1 Following

Follow

My focus within Information Technology is API Management, Integration, and Identity—especially where these three intersect.

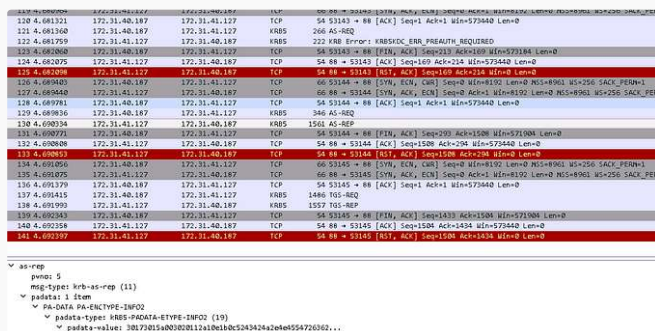
No responses yet



What are your thoughts?

Respond

More from Robert Broeckelmann





Robert Broeckelmann

Kerberos Wireshark Captures: A Windows Login Example

This blog post is the next in my Kerberos and Windows Security series. It describes the...

May 17, 2018



254



4



Robert Broeckelmann

OpenID Connect Logout

The OpenID Connect (OIDC) family of specs supports logout (from a single application)...

Jul 12, 2017



490



5



Robert Broeckelmann

HTTP POST vs GET: Is One More Secure For Use In REST APIs?

The use of HTTP POST vs HTTP GET for read-only (or query) operations in REST APIs...

Feb 6, 2021



78



Robert Broeckelmann

Authentication vs. Federation vs. SSO

Authentication. Federation. Single Sign On (SSO). I've mentioned these concepts many...

Sep 24, 2017



859




6

[See all from Robert Broeckelmann](#)

Recommended from Medium

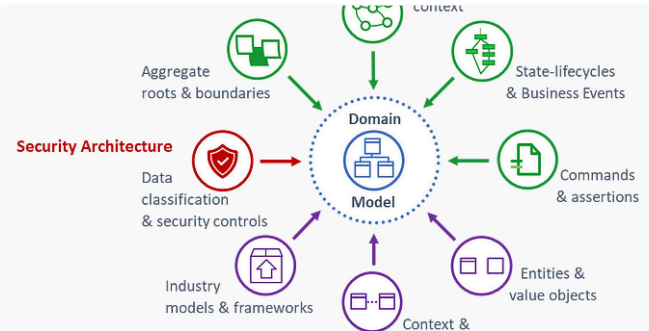



 In Stackademic by Crafting-Code

I Stopped Using Kubernetes. Our DevOps Team Is Happier Than Ever

Why Letting Go of Kubernetes Worked for Us

★ Nov 19 🖱 3.5K 💬 113 📌 ⋮



 In API Central by TRGoodwill

API Design Practice

A practical guide to API QA and the design of stable, coherent and composable business...

May 9, 2023 🖱 1K 💬 11 📌 ⋮

Lists



Coding & Development

11 stories · 926 saves



Company Offsite Reading List

8 stories · 170 saves



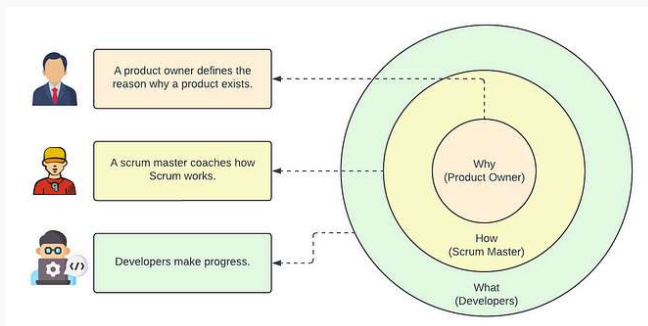
data science and AI



40 stories · 296 saves

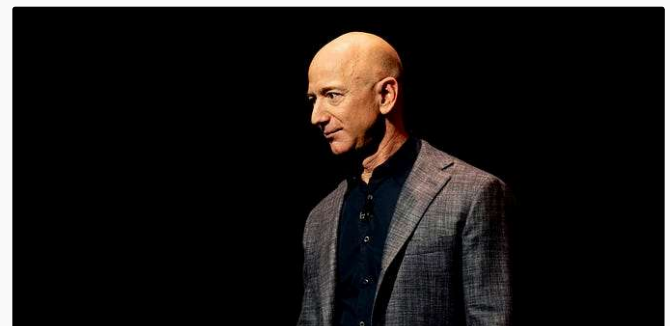



Natural Language Processing

1841 stories · 1463 saves



 In Beyond Agile Leadership by Eiki Takeuchi 

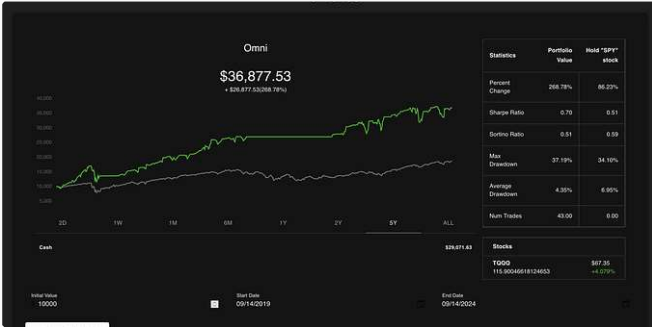


 Jessica Stillman

Why Hiring High-Performance Developers is the Biggest Mistake...

When I was working as a software engineer in Japan, I saw numerous smart and talented...

Nov 11 368 32



In DataDrivenInvestor by Austin Starks

I used OpenAI's o1 model to develop a trading strategy. It is...

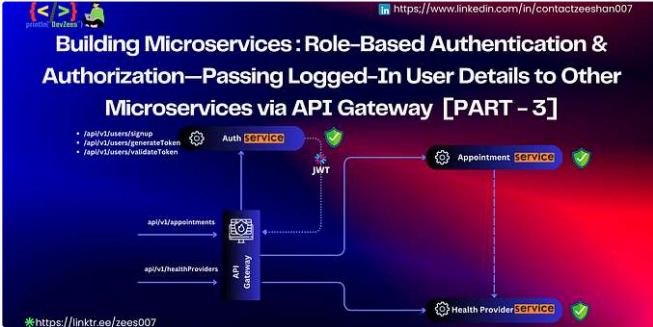
It literally took one try. I was shocked.

Sep 16 6.8K 174

Jeff Bezos Says the 1-Hour Rule Makes Him Smarter. New...

Jeff Bezos's morning routine has long included the one-hour rule. New...

Oct 30 13.7K 322



In Level Up Coding by Zeeshan Adil

Building Microservices [PART-3]: Role-Based Authentication &...

Welcome Back to DevZees

Nov 12 106

See more recommendations