software AG
GOVERNMENT SOLUTIONS

API Management for a digital world

Table of contents

White paper

The rise of the modern API

Ever since the dawn of the Internet, people have struggled with how to get one computer to talk to another. Early business systems had no provision for such interactions. They were entirely closed worlds unto themselves.

As organizations set up early networks, the question of how to get applications to interact with each other became a pressing concern and led to the introduction of Remote Procedure Calls (RPCs). A client computer might directly interact with the program running on a server or host by calling procedures (or subroutines, methods, etc.) over the network.

As long as every host system had its own proprietary architecture, operating system and programming environments, however, such RPCs were rarely practical. As those systems matured, a level of standardization became more commonplace, and the Application Programming Interface (API) was born.

Early APIs exposed RPC interactions, and the fact that we were able to define a software interface allowed us to abstract the underlying code–thus simplifying the software on the client side and providing some measure of flexibility.

Flexibility, however, is a relative term. For many years, virtually any change in the code on the server required a commensurate update on each client–a problem we now call tight coupling. Organizations were able to live with tightly coupled client/server applications. After all, they had little choice.

That is, until the Internet came along.

Thanks to simple protocols like HTTP and HTML, browser interactions with web servers were loosely coupled. Any browser worked with any Web page on any Web server and wouldn't break when someone updated the page–more or less.

But what about application-to-application integration? How do we make those interactions as loosely coupled as the Web?

The answer: Web services. Take the suddenly popular eXtensible Markup Language (XML) as a basis, combine it with the hard-won lessons of interface publication and discovery from the Common Object Request Broker Architecture (CORBA®) and hammer out standard protocols for contracting software interfaces–now called services–and the formats of the messages they exchange.

Oh, and don't forget publication and discovery of the services, which require more than an XML-based standard. We actually need architecture. And lo, Service-Oriented Architecture (SOA) was born, like a phoenix from the flames of CORBA.

The service contracts that defined Web services provided looser coupling as compared to the earlier generation of client/server APIs but required traditional middleware in order to support the data transformation and content-based routing at a scale that enterprises required to implement SOA in practice.

These Enterprise Service Buses (ESBs) were an important part of the early SOA story to be sure. But the combination of ESBs and Web services proved difficult to implement in practice–and with the advent of cloud computing, this first-generation SOA story became ripe for further innovation.

Today, the principles of SOA have become a part of the fabric of enterprise distributed computing, and by navigating the gauntlet of Web services, we have reinvented the API for today's cloud and mobile-centric world.

REST and the need for API management

Early in the development of Web Services, it became clear that middleware alone was insufficient to ensure services complied with their contracts, as well as security and other requirements in the production environment. As a result, Web services management became a nascent but growing market.

Over time, Web services management transformed into SOA management, then run-time SOA governance, and now API management (although Gartner uses the term application services governance).

After all, Web services are a type of API, and managing them is an aspect of governance. And yet, the real reason for this shift in terminology is the rise of Representational State Transfer (REST).

Today, our APIs are more likely to be RESTful, HTTP-based interfaces than SOAP-based Web services. In fact, perhaps the most successful part of REST to date has been the simplification of the API.

We no longer need a language-specific protocol that depends upon sophisticated network controls under the covers. Today we can take HTTP for granted, and a simple request to a URL suffices to establish any interaction we care to implement between any two pieces of software we like, regardless of language.

REST, however, isn't perfect. It's rather vague about many of the specifics of its implementation, leading to widespread confusion. For numerous vendors, of course, confusion means opportunity, and the API management market is exploding as a result.

You'd think as the API story shifted over the past decade from Web services to RESTful interfaces that the core challenge would have become simpler. In fact, REST's inherent flexibility has actually exacerbated the management headache–raising the bar on API management.

The API economy

The enterprise integration story, dating from early RPC-based efforts up through first-generation SOA, all focused on integration internal to the enterprise. Today, enterprise integration is as important as it ever was. But we're also living in a mobile and cloud-enabled ecosystem, where connecting to citizens, customers, partners, suppliers, or anyone else is an essential part of the digital fabric of government and business.

The role of the API, unsurprisingly, has also expanded, as APIs are the glue that holds this digital ecosystem together. They are as important for the consumers of software capabilities as the providers, and they are now the new distribution channel for products and services as well.

Organizations of all sizes now use APIs. Thousands of app developers out there can innovate using your APIs, just as you can innovate using theirs.

Enterprise organizations who may have never thought of themselves as offering software-based products or services are now able to leverage APIs to expand their offerings. As enterprises in multiple industries become software-driven organizations, APIs become the means for providing value to citizens, for maintaining efficient relationships with suppliers, and for participating in the broader commerce communities to which they belong.

Correspondingly, API management takes on new importance in this API economy. Security is basically table stakes. Managing the performance of individual APIs also becomes a priority. And the interactions surrounding API discovery and usage, including the monetization of APIs, all become a part of the API management equation.

How to develop an API strategy

- Take stock of the application and data assets that would be worthwhile to expose as APIs. If you have already implemented services as part of an SOA initiative, you're ahead of the game.
- Decide which APIs are for external consumption vs. APIs for internal use. For externally facing APIs, hammer out the appropriate business model. Who will access them?
- Implement API management – the tooling as well as the appropriate policies and procedures for supporting your goals as well as governance priorities like security and compliance.

In fact, API use cases are simple extensions of Web, mobile, and cloud-based apps. All such digital interfaces require their own life cycles. Management, governance, and security are absolutely mandatory in this API-enabled app world.
The governance of APIs, in fact, extends the hard-fought lessons of SOA governance: If we represent policies in a metadata format, then our technology can automatically manage and enforce such policies. It is therefore possible–and in fact, desirable–to automate operational policies, such as access, authentication, credential validation, and Service Level Agreements (SLAs).

Today's API management, furthermore, extends these basic SOA governance capabilities to policies for API monetization, run-time consumption and monitoring, consumer adoption, and traffic limit enforcement.

As enterprises increase their participation in the API economy, they eventually realize that API management must rise above the technical details, as APIs become a critical business tool in themselves. The real value in API management, therefore, is managing business itself.

APIs: The glue of digital transformation

The word digital has been with us for years. But today it's experiencing a renaissance, as enterprises in every industry reinvent themselves as software-driven organizations. And yet, digital paradoxically isn't about software. Digital is all about the consumer.

The only reason technology is central to the digital movement is because more than ever before, customer preferences and behavior are driving enterprise technology decisions. Customers (citizen consumers as well as business-to-business) demand diversity in their technology touch points, ranging from computers to televisions smartphones to wearable devices, and more.

The consumer experience, however, doesn't end with the device; it begins with it. For enterprises to meet the diverse and dynamic needs of their customers, they must connect the dots between user interface and back-end technology, between front-office and back-office. Furthermore, this end-to-end technology story must perform–every time, in real time. Consumers demand no less.

Essentially, digital transformation represents an end-to-end rethink of what it means to put an application into the hands of consumers. Instead of the monolithic or tightly integrated applications of the past, today's digital app is more likely to consist of multiple components, scattered across the cloud.

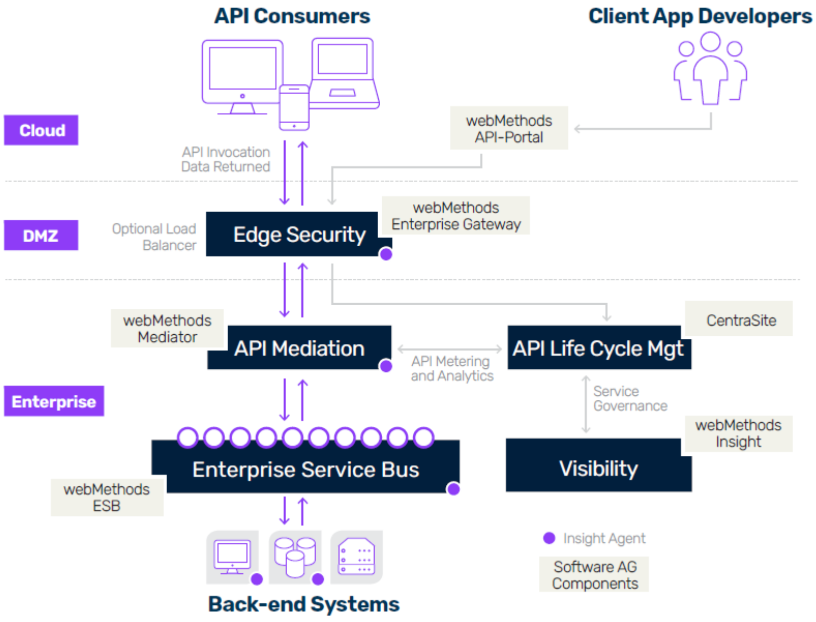
And the glue holding such applications together is the API.

As with any glue, APIs essentially stick things together. On one side, developers provision mobile, Web and cloud applications with Web APIs. On the other side, developers (who are often third-party coders) create new apps and mashups using open APIs.

Of course, in many cases, both the consumer and provider of an API reside within the same organization. In such cases, developers work to compose back-end services and expose such compositions as easily consumable APIs.

At this point, such organizations must implement API life-cycle management, mediation, and an ESB or other intermediary that can handle message transformation, content routing and protocol switching. This diagram shows how Software AG has architected its API management capabilities in concert with other elements of its middleware suite.

Many companies also roll out an API portal for third-party developers to access. At the least, such a portal should include the ability to self-register. In other cases, it's a full-blown marketplace, with e-commerce transaction capabilities, recommendation engines, and the like, depending on the monetization model for the APIs.



The API Management Environment (Source: Software AG)

It's also essential to consider how APIs impact the software life cycle. API testing capabilities, documentation, code samples, API versioning and the ability to support collaboration are all part of the broader API management challenge.

Finally, managing the scalability of APIs is also an important aspect of API management. Such scalability management capabilities can also be a critical part of the organization's protection against denial-of-service attacks.

The strategic business value of APIs

For any API strategy to be successful, organizations must manage the entire process of designing, developing, deploying, versioning and retiring APIs. Security is essential as well: API management must provide authentication, mediation, payload transformation and connection to supporting back-end services and systems in order to secure all digital apps.

Visibility into API usage is also an essential part of API management, as organization must collect and leverage metrics for performance dashboards, SLA violations and invoicing, when appropriate.

Supporting the development life cycle is also critical, including the ability to design, develop, deploy, virtualize, and publish APIs. Organizations should leverage a single asset catalog in order to keep track of APIs as well as services, schemas, documents and other related assets in one place.

Finally, API management should also focus on monetization. APIs are more than software interfaces. They can easily become the products in enterprise software marketplaces, adding both community and revenue to the bottom line.

Never forget that APIs are the glue that holds a digital transformation initiative together. And while mobile and the rest of the user experience part of the digital story gets most of the attention, the enterprise context for digital is end-to-end.

About the author

Jason Bloomberg is the leading industry analyst and expert on achieving agile digital transformation by architecting business agility in the enterprise. He writes for Forbes, Wired, Cortex and The Agile Architecture Revolution. As president of Intellyx, he advises business executives on their digital transformation initiatives, trains architecture teams on Agile Architecture, and helps technology vendors communicate their agility stories. His latest book is The Agile Architecture Revolution (Wiley, 2013).

ABOUT SOFTWARE AG GOVERNMENT SOLUTIONS

Software AG Government Solutions is a market leader in best-in-class commercial software dedicated to serving the U.S. Public Sector and A&D supplier community. Our solutions remove the barriers of connecting data from every source – from cloud to edge – across an extended enterprise. We are recognized by Gartner and Forrester as leaders in API Management and Hybrid Integration (webMethods), Industrial IoT (IIOT) platforms (Cumulocity), and IT Portfolio Management and Enterprise Architecture (Alfabet). Widely embraced by our customers, we leverage a "Prove IT First and Prove IT Fast" approach to solving critical IT challenges. This approach has proven to minimize risks associated with IT investments while allowing us to prioritize building trusted relationships with our customers.

software AG
GOVERNMENT SOLUTIONS