CITO Research
Advancing the craft of technology leadership

May 2012

Enterprise-class API Patterns for Cloud & Mobile

Sponsored by intel

McAfee
An Intel Company

Contents

CITO Research
Advancing the craft of technology leadership

Foreword: The API Is the New Website

by John Musser, Founder, ProgrammableWeb

We live in an app economy. Increasingly, the primary digital mode of engagement between businesses and their customers, partners, and even employees is through apps. To deliver services via apps, most businesses develop application programming interfaces (APIs) that support machine-to-machine interactions over the Web.

Programmable Web has determined that there are more than 5,000 APIs available on the Web today, up from near zero in 2005. Growth has been exponential (Figure 1).

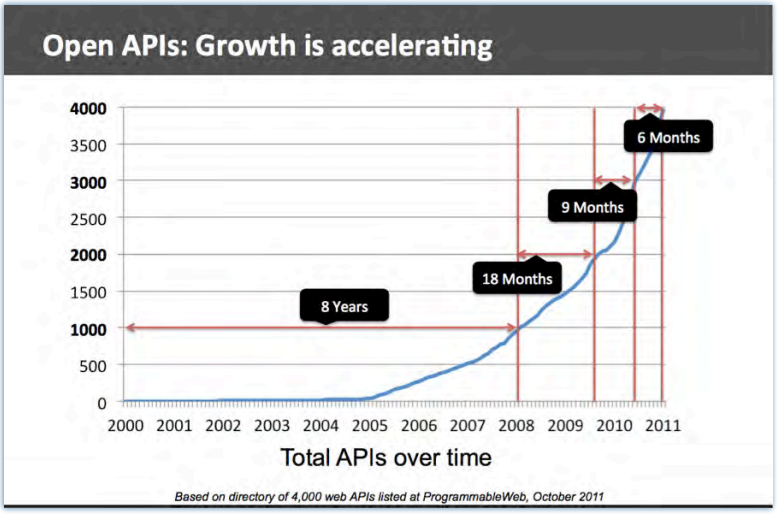

Figure 1. APIs: Growth is accelerating (Source: Programmable Web)

In many ways, APIs are the websites of this generation. In 1995, businesses were asking, "Why do I need a website?" By 2000, every business needed to say, "Of course we have a website." In 2007, everyone was asking, "What is an API and why do I need one?" Now, the pressure is on to say, "Of course we have an API!" Not only will every company eventually have an API; many of them will soon have multiple APIs for different purposes.

Of the dozens of categories and thousands of APIs tracked by ProgrammableWeb, one of the fastest growing categories is the enterprise category—businesses communicating with each other. One of the main reasons is the rapid adoption of cloud and cloud-based technologies within the enterprise, whether it's through the CIO's office or through the side door. Cloud solutions are primarily integrated through APIs. Think of APIs as the "glue of software as a service (SaaS)." APIs are the way that data gets into and out of the cloud, and from enterprise to enterprise.

It's only now that enterprises are learning best practices for running and securing APIs. It's not enough to offer an API; it needs to be reliable, scalable, and secure. Many enterprises don't really know how to offer APIs with the same security and service level as their enterprise applications.

Until now, the use of APIs grew organically at the department level on an app-by-app, partner-by-partner basis, with no enterprise IT standards or architecture enforcement. This has not changed, even as most companies have adopted APIs.

The vast majority of developers today are trained and working in a lightweight, simple architectural style called REST (Representational State Transfer), which is ideal for the biggest sector of the app market, mobile devices. But many of the first-generation APIs were written using SOAP (Simple Object Access Protocol) or other variations on web services. Enterprises offering APIs will need to "speak" REST to the outside world in order to continue to grow (Figure 2).

CITO Research
Advancing the craft of technology leadership

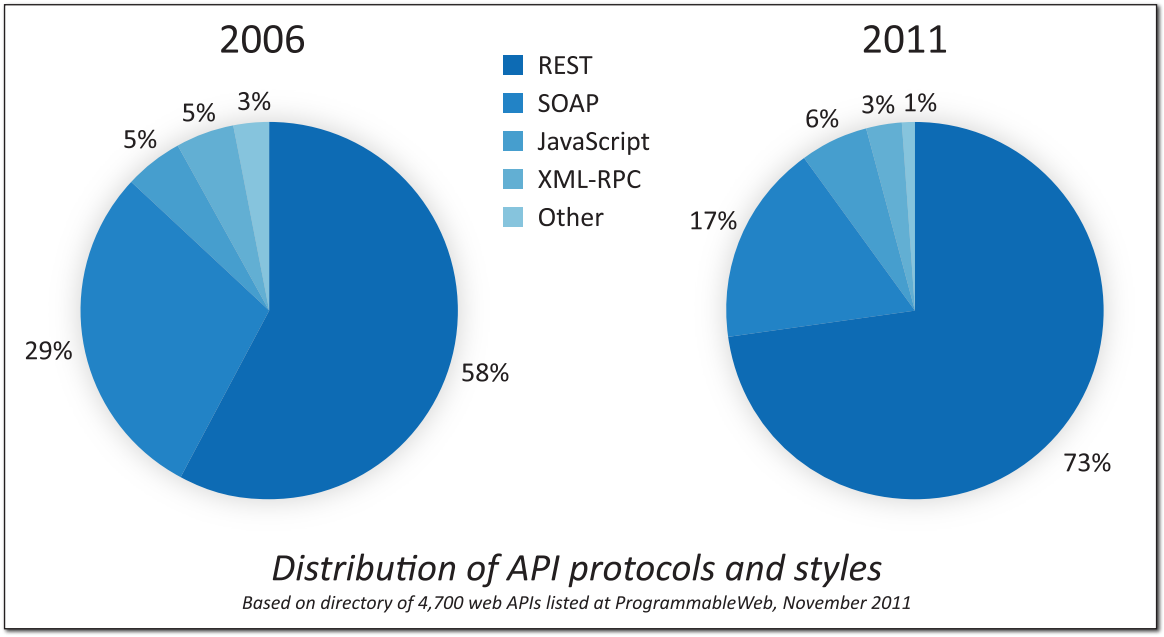

Figure 2. REST versus SOAP: Simplicity wins again (Source: Programmable Web)

Requirements for Enterprise APIs

This CITO Research paper, with insight from Dan Woods, coauthor of *APIs: A Strategy Guide* (O'Reilly Media, 2011), describes several API usage patterns. By adapting these patterns, enterprises will be able to follow best practices in API security. Before covering these patterns, this section describes enterprise architecture requirements for designing, building, managing, and protecting APIs.

Standards for Consuming and Exposing APIs

In the early days, most APIs were built based purely on SOAP standards. More recently, multiple standards have emerged, such as REST and JSON (JavaScript Object Notation). Additionally, enterprises need to mediate between these standards and design patterns on the back end of their technology stack, which is not changing as quickly as the front end. Today, a typical organization works with between 5 and 15 cloud providers. Any department could write a connection to consume a cloud service such as Amazon Web Services, Google Apps, or Salesforce.com. Here are just a few of the problems with this approach:

- Security is commonly hard-coded per provider. The security protocols must be compatible for each relationship. Each new hole in the enterprise security perimeter increases exposure to threats and vulnerabilities. Also, many providers do not provide any security, or only minimal security, at the network level. It is hard to find a provider that can support application-level security that meets enterprise-grade needs.
- Each department maintains different financial, SLA, and technical support agreements with each provider, which results in engineering building one-off solutions that do not share infrastructure. Maintaining multiple one-off solutions, with no common standards, becomes a major headache.
- Auditing is problematic, with no easy way to identify redundant and critical services.

- The more APIs you consume, the more you have to worry about management and compliance. Compliance is a major issue if you are an insurance, healthcare, financial, or banking company. You need to implement proper policies for exposure, consumption, and availability of services based on the standard that you need to comply with (such as HIPAA, SOX, COBIT, PCI, and so on).
- Exposing an internal application may require mediation with existing infrastructure, which is not necessarily primed for communication on the open Internet.

Individual applications are exposed on app servers that may not be security-hardened or scalable. The mix of technologies in a large enterprise, from mainframe applications to web servers, all speak different languages and use different transport and security protocols (see Figure 3). Commonly, management of API keys presents the single biggest challenge for IT.

Enterprise-Class Security

Enterprise-class security includes authentication, authorization, and perimeter defense.

Authentication in APIs provides a way to ask the question: "Is the entity making this request really who they say they are?" **OAuth** is the predominant API authentication model, used especially by organizations with APIs associated with websites or mobile applications that require passwords.

By creating a token that exists only during application handshake, OAuth prevents password propagation around the Internet. It is best suited for large groups of unknown developers trying to access your API. It is also lightweight and relatively easy to use; "new school" developers are quickly becoming fluent in OAuth.

Enterprise developers may be more familiar with authentication mechanisms such as **X.509 certificates, Security Assertion Markup Language (SAML)** and

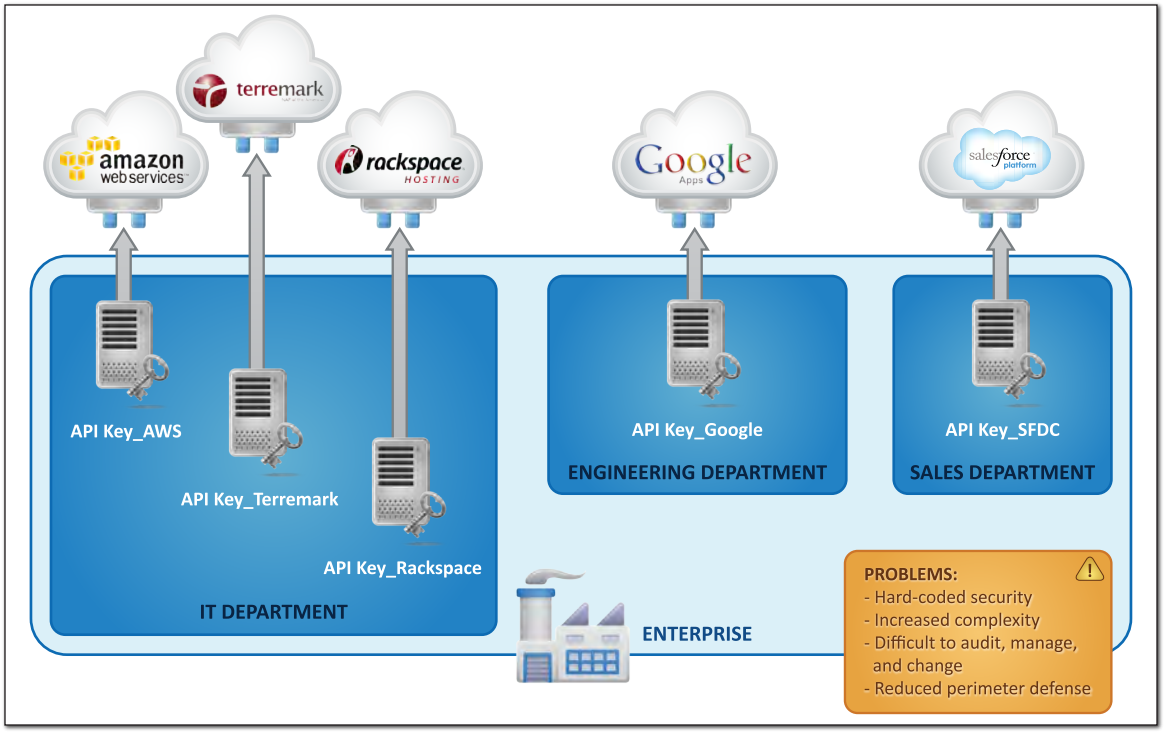

Figure 3. Consuming APIs: A Wild West Approach

two-way **Secure Sockets Layer (SSL)**, which are implemented as part of web service standards colloquially known as WS- *.

However, an authenticated client can still pose a threat. Attacks such as denial-of-service, code injection, malware, and data leaks must all be accounted for when designing an enterprise API strategy. Tactics such as data validation, input validation, pattern-based scanning, heuristics, antivirus scanning, and malware scanning can mitigate these threats.

Authorization provides a means of asking, "Is this entity allowed to do what it is attempting to do?"

The most prominent API authorization model is XACML (eXtensible Access Control Markup Language), which decouples authorization from the point of use (in other words, the application itself). With XACML, authorization policies can be updated on the fly and affect all clients immediately.

Perimeter defense must also be considered. Every time your data crosses the edge of an application, enterprise firewall, or the cloud, a vulnerability is exposed. Enterprise, cloud, and application edge security must be built into any enterprise API scheme, as it provides the earliest form of detection.

Visibility and Control

Most enterprises lack overarching visibility into and control of APIs. Departments often provision their own computing services from the cloud, frequently provided via APIs. Typically, these services include the following three major categories:

- **Infrastructure as a Service (IaaS):** Cloud providers offer physical and virtual machines, storage, firewalls, load balancers, and network connections based in large data centers. The cloud user is usually responsible for patching and maintaining the operating systems and application software. Providers bill IaaS services on a utility basis—cost reflects the amount of resources consumed.
- **Platform as a Service (PaaS):** The cloud provider delivers a computing platform that includes an operating system, programming language execution environment, database, and web server. In some offerings, resources scale automatically to meet demand.
- **Software as a Service (SaaS):** Application software is installed by cloud providers and cloud users access the software remotely. This "hands-free" approach means the software can scale and apportion itself according to demand without intervention by the user. Typically users do not manage the infrastructure or platform.

SaaS and PaaS providers are circumventing traditional enterprise architecture. Compliance and visibility has decreased. Simply put, your enterprise is likely already part of the app economy. The question is, how are you managing your API traffic? Do you have a control point to manage that participation?

Enterprise APIs are not science projects; they're conducting enterprise-class business and require enterprise-class visibility and control. What path can enterprises take to prepare for secure use of APIs?

CITO Research
Advancing the craft of technology leadership

The API Gateway: A Control Point for the App Economy

The path to security, reliability, and scalability comes through an API gateway design pattern, typically instantiated by a network edge gateway software appliance, sometimes called a security or service gateway. An API gateway acts as a control point between enterprise IT infrastructure and the outside world accessed through APIs, including the cloud. Functions of API gateways may include:

- **Location virtualization:** API Gateways allow you to invoke services that run anywhere, any time and allow you to seamlessly move your services around at will without affecting your infrastructure.
- **Version control:** API gateways provide API version control and abstraction.
- **Centralized security and policy enforcement:** API gateways centralize security and policy enforcement, without having to hard-code security into individual applications.

Gateways offer security architects and developers flexibility for custom coding where needed and also provide a way to apply security policy, routing, services, or out-of-the-box security functions such as encryption and authentication.

- **Global event visibility:** Gateways can improve visibility with dashboards that facilitate an understanding of incidents across APIs, such as coordinated malware attacks. Gateways can be designed to track API service call activity for security event reporting or chargebacks. Since they intermediate all transactions as a proxy, API gateways supply rich audit trails, coordinated analysis, and detailed monitoring across previously siloed and separate department implementations. Security Information and Event Management (SIEM) capability is also provided, providing real-time analysis of security alerts generated by network hardware and applications.

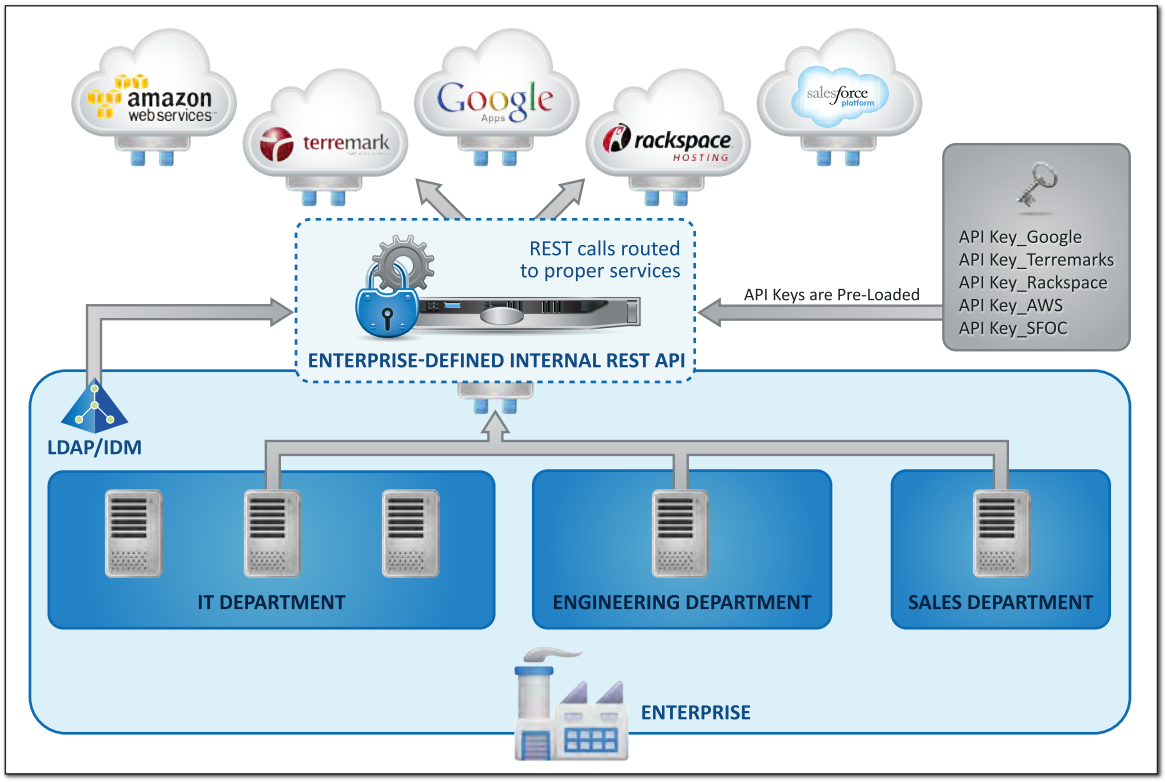

Figure 4. API gateways serve as an intermediary between the enterprise and external APIs

CITO Research
Advancing the craft of technology leadership

Cloud Service Brokers: A Role that Simplifies Consumption of Cloud APIs/Services

IT is evolving from an implementer of point-to-point integration and infrastructure for cloud services into an orchestrator of services consumed by internal department-level developers. Today, your organization might be managing cloud services from a dozen providers. Tomorrow, when this number explodes, point-to-point management of all of these services, including billing, governance, security, and authentication, will not be a viable option.

Use of an API gateway delivers a key technology enablement platform so that your IT department can become what Gartner Group calls a "cloud service broker" (CSB). Additionally, many third-party companies are becoming CSBs themselves to package and deliver services or integrations for vertical industries. Analyst consensus estimates that by 2015, 20% of all cloud services will be intermediated by CSBs. A cloud service broker is an emerging role for IT or intermediaries, and an API gateway is the centerpiece technology platform that enables this usage model.

Gateway Usage Patterns

An enterprise API gateway can be advantageous in many contexts. This section describes several usage patterns.

Supporting Partners with APIs and Web Services

Business usage model: API gateways are most frequently used to ensure that two businesses have as frictionless an interaction as possible. To support automated exchange of information and process integration, many companies expose assets in the hopes of drawing in more revenue, expanding their partner bases in the process. One well-known use case is Netflix's integration with more than 200 hardware devices, from Roku to Xbox, vastly expanding its subscriber base.

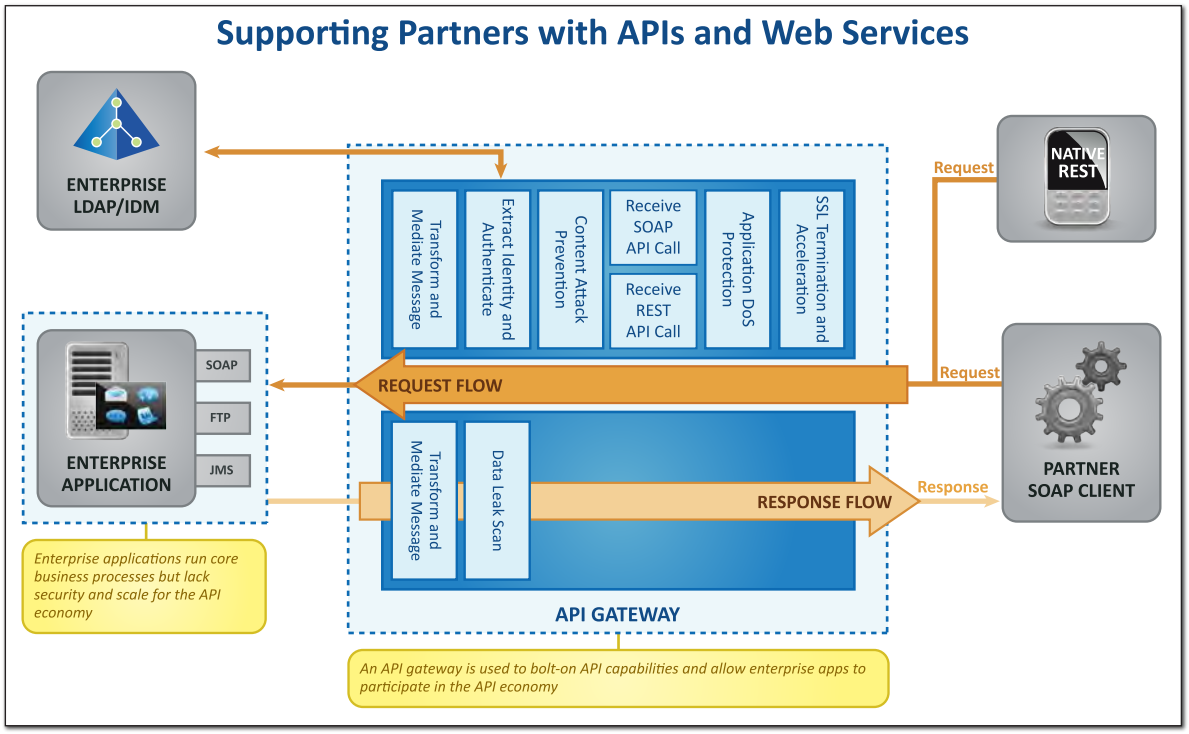

Figure 5. Supporting Partners Pattern

CITO Research
Advancing the craft of technology leadership

To successfully expose assets, legacy protocols need to be translated. Attempting to do this sequentially, partner-by-partner, seriously inhibits scalability and time-to-market.

API gateway solution: The API gateway mitigates risk by offering enterprise-edge security. In this model, the REST API is exposed on the gateway, which can be on-premise in a secured DMZ or a virtual appliance in a trusted network. REST requests are translated to SOAP so that backend applications don't have to be rewritten. Security is enforced at the gateway, not on the backend.

Supporting Mobile Use of Enterprise Services

Business usage model: In order to reach consumers (and increasingly, partners and employees), you need mobile apps. A mobile API, however, can open up access to enterprise systems, which creates a security problem. Via mobile device, anyone could send a function call directly into your enterprise. Worse, mobile devices are mass-market devices, multiplying the number of potential security holes very quickly. This contrasts traditional partner integration, where partners were a mere handful.

Mobile devices can bring potentially millions of messages a day from thousands of devices. You need a control point to handle traffic and minimize impact on backend application servers—a type of "mobile middleware."

API gateway solution: The gateway maintains security and manages volume and complexity as you expose enterprise applications via mobility. The gateway manages perimeter defense, scanning messages for malicious content. It also manages changes to mobile platform APIs centrally. The gateway speaks REST and JSON, which, combined with the volume of API calls, demands a responsive, scalable platform that can serve requests while maintaining the integrity of enterprise apps. In this use case, the gateway authenticates the client and obtains authorization using OAuth. API responses are typically sent as a JSON payload for native mobile appli-

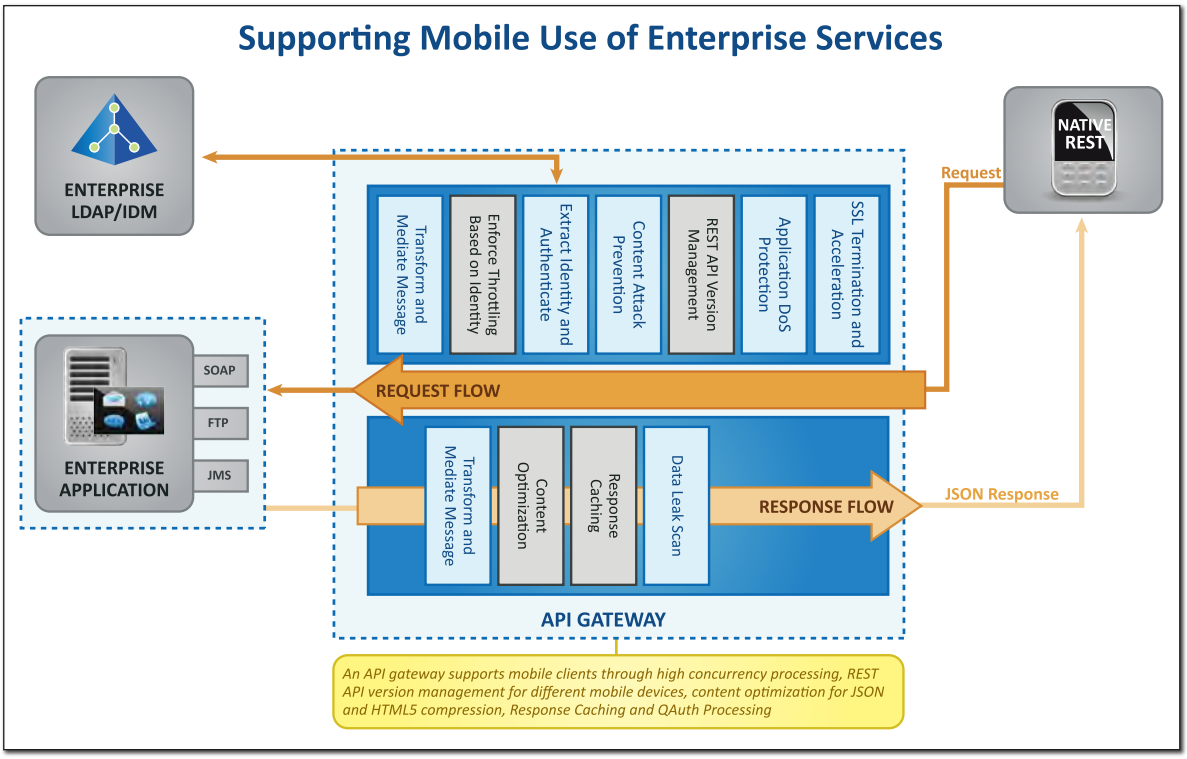

Figure 6. Mobile Use Pattern

cations, which interpret them into an object for use in the native mobile application programming environment.

Controlling Use of Cloud Infrastructure as a Service (IaaS)

Business usage model: With service providers such as Rackspace or Amazon Web Services, your organization runs virtual machines (VMs) on hardware in third-party data centers. Savings accrue because you can run applications on rented computing power without paying for on-premise resources. To offer more direct and rapid integration, these providers offer VM control via HTTP-based API calls rather than through a web console. Simply put, they offer API controls for virtual machines. These APIs allow administrators to apportion the infrastructure devoted to a given enterprise application with simple, easy to use REST function calls.

API gateway solution: The gateway lets organizations take advantage of API controls for running VMs in the cloud. The configuration of, and access to, cloud-based infrastructure can be managed from a central place. The gateway can also act in a client capacity by securing and mediating these REST API calls. The API gateway provides a façade or standard abstraction layer for developers. This allows the enterprise to easily switch between multiple cloud IaaS providers without expensive rip-and-replace of custom code.

The gateway can prevent VM creations or deletions from rogue users or unauthorized users. Role-based rules can help manage the sprawl of multiple VMs in the cloud for cost savings and provide audit controls on external VM usage. Further, the gateway can interface with your enterprise LDAP, extending existing roles, and access control policies for provisioning new VM infrastructure, without creating a need to twist your existing identity system to meet the whim of the cloud provider.

Without a gateway, setting up rules about who can set up or terminate a VM in the cloud becomes a custom development project, done ad hoc, per provider. Terminating a VM might lead to the loss of a vital business service.

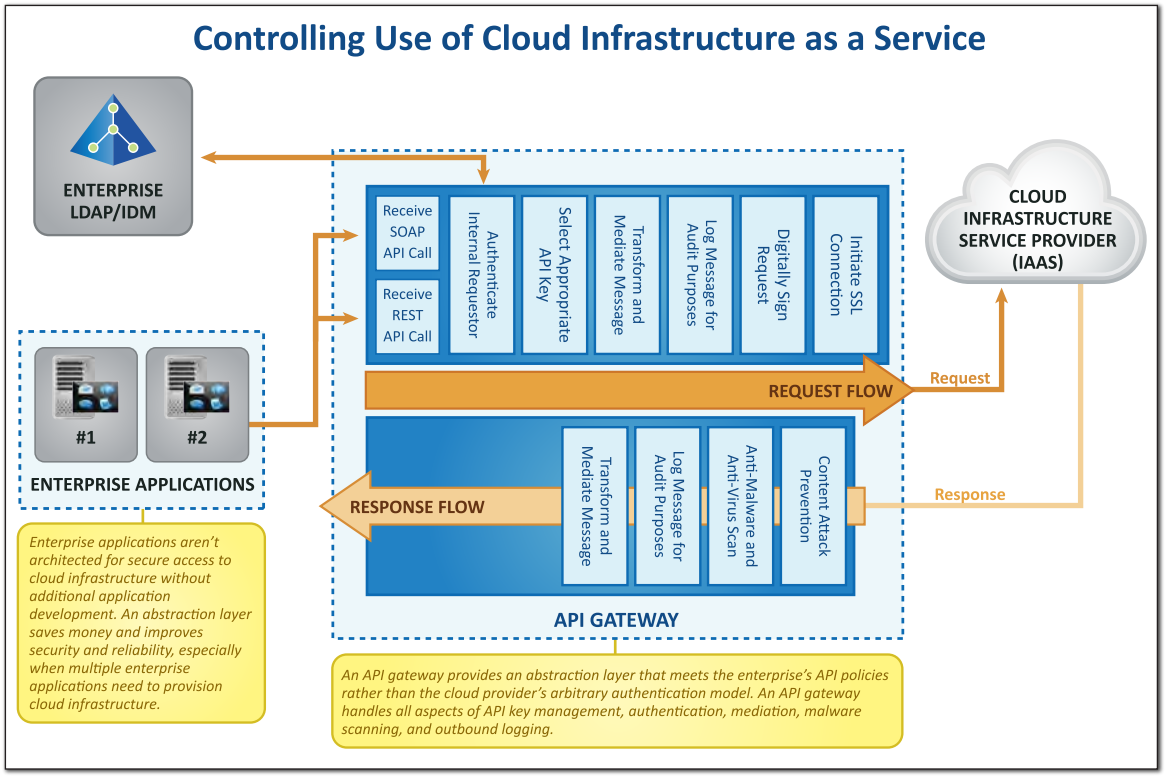

Figure 7. Infrastructure as a Service Pattern

Adding a raft of unauthorized VMs to your cloud wastes money. Without a gateway, IT must hard-code authentication tokens into every application it hosts in the cloud, a barrier to the promised savings and efficiency of cloud computing.

Controlling Use of Platform as a Service (PaaS)

Business usage model: Many enterprises are building software as a service (SaaS) apps themselves, using PaaS providers such as Heroku, Force.com, and Engine Yard. A gateway can secure, propagate, and manage these apps. Building SaaS apps on PaaS providers' platforms implies a heavy reliance on the providers' security controls, which are generic to attract the widest customer base, but often weak. There is no guarantee that enterprise policies will work in a cloud-based app built on these platforms with their given toolsets.

API gateway solution: Using an API gateway for PaaS-created applications offers better authentication and encryption protection, including OAuth, SSL, shared secrets, or signatures. A gateway in a virtual form factor allows you to put your enterprise security policies in the cloud.

The gateway applies policies to data as it enters a hosted application or database, so your enterprise policies can be protected while still gaining the advantage of a cloud-based deployment.

API gateways are available as cloud-based virtual machines, which means the same cost savings offered by the cloud provider can be extended to the gateway itself—and it will be operationally closer to the cloud assets you create.

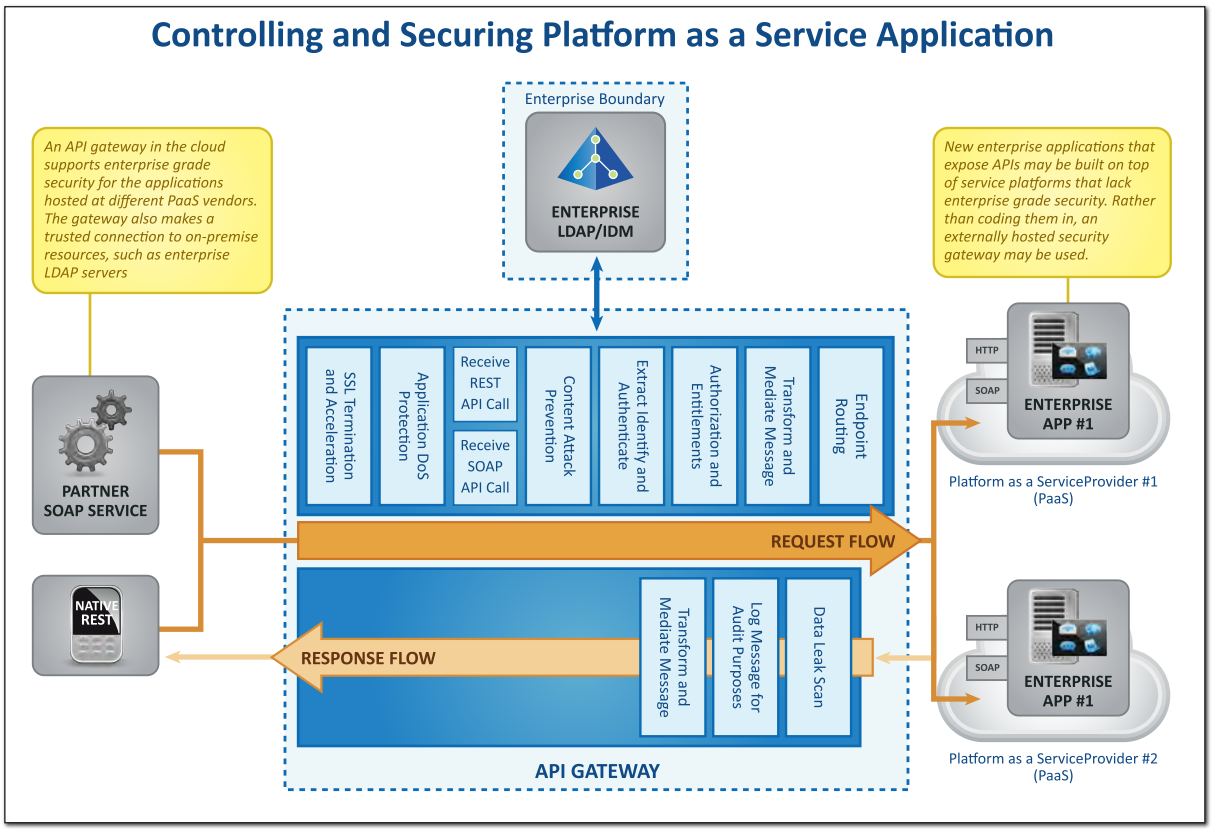

Figure 8. Platform as a Service Pattern

CITO Research
Advancing the craft of technology leadership

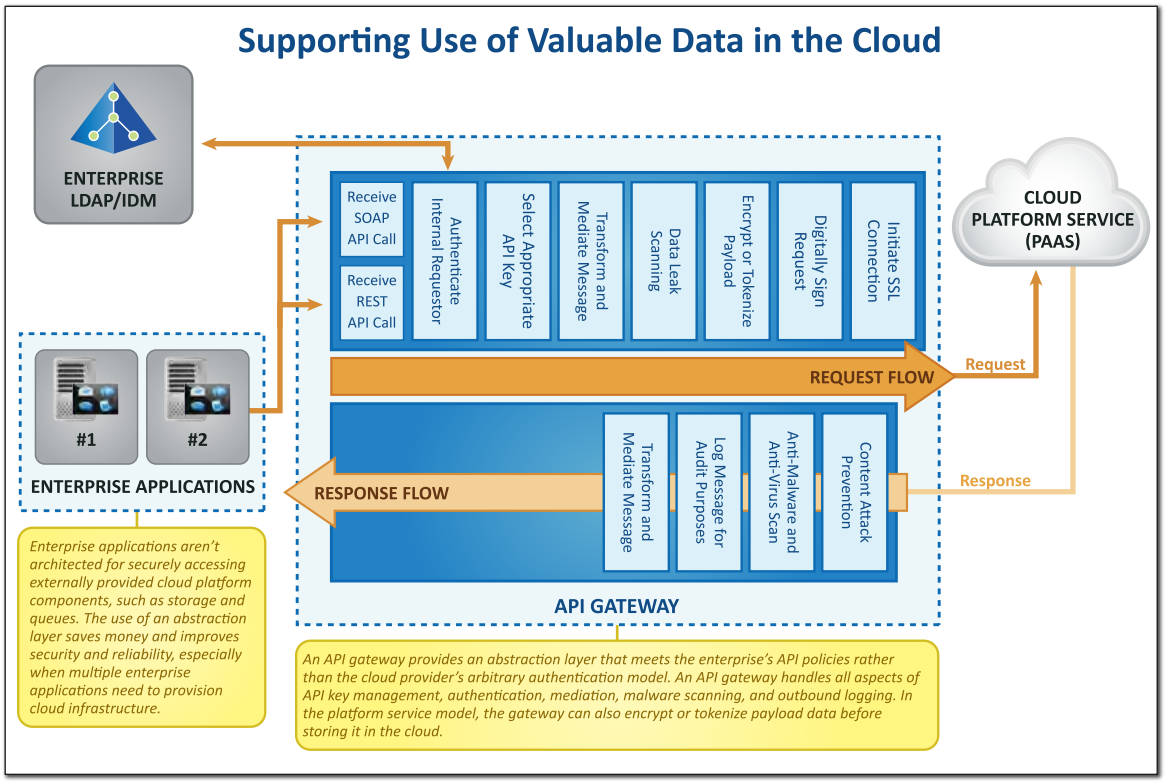

Figure 9. Data in the Cloud Pattern

Projecting Security Policy on Data in the Cloud

Business usage model: Many enterprise applications now use data from a combination of internal databases and data from cloud-based PaaS providers, such as Dun & Bradstreet, Data.com, and Salesforce. When data is split between the cloud and the enterprise, business risk is increased because core business systems are exposed. It's vital that security and control mechanisms for externally hosted data are as robust as those for on-premise data. The enterprise application needs to authenticate itself to the cloud service using a method such as API keys or shared secrets. Each cloud service is likely to use a variant of several of these methods.

API gateway solution: The API gateway can manage sharing of sensitive data between the enterprise and cloud-based PaaS providers. The gateway can encrypt or tokenize portions of the data, such as personally identifiable information (PII) or intellectual property. With encryption, the enterprise keeps the keys and can determine how to protect data before it is sent to the cloud.

The gateway can keep a copy of the API call before it's sent, in case the data payload doesn't come back, by caching a copy in a secure area or database under the enterprise's control.

The gateway can abstract the interface to the cloud services so that authentication can be managed in a standardized way, regardless of the combination of methods used by the cloud service.

Securing Cloud Applications (Software as a Service)

Business usage model: Self-service is one of the biggest trends in today's enterprise. Many popular business services are offered in the cloud. However, resolving security issues associated with these services may neces-

CITO Research
Advancing the craft of technology leadership

sitate expensive development on the provider side. The multiple relationships between business users in departments have become a management nightmare for IT, which has no practicable way to enforce enterprise policies in cloud-based apps hosted by third parties.

API gateway solution: The API gateway is a valuable architectural addition when Enterprises or Service Providers are building their own SaaS applications. It lets you scale your SaaS applications more easily while protecting a wide range of clients, internally and externally. Alternatives to this architecture place more software development, security and performance burdens on the web server and application server tier, which are really designed for end-user communication, not API transactions..

The gateway offers advanced, out-of-the-box security features to improve the security posture provided by SaaS operators, which often lack these features. The built-in security of a gateway saves tremendous amounts of money and resources that would otherwise be spent on patching the security holes left by SaaS providers.

An API gateway is a critical component that extends the standard three-tier architecture for modern SaaS applications. Without a dedicated gateway, API channels and their accompanying security controls such as OAuth, SAML, SSL, X.509 processing, or shared secrets become a custom coding project on top of the application server. Worse, complex heuristics for malware scanning and data leak protection cannot be easily added. Large SaaS applications that require security and enterprise-grade security may benefit from a security gateway for reduced coding time and increased performance.

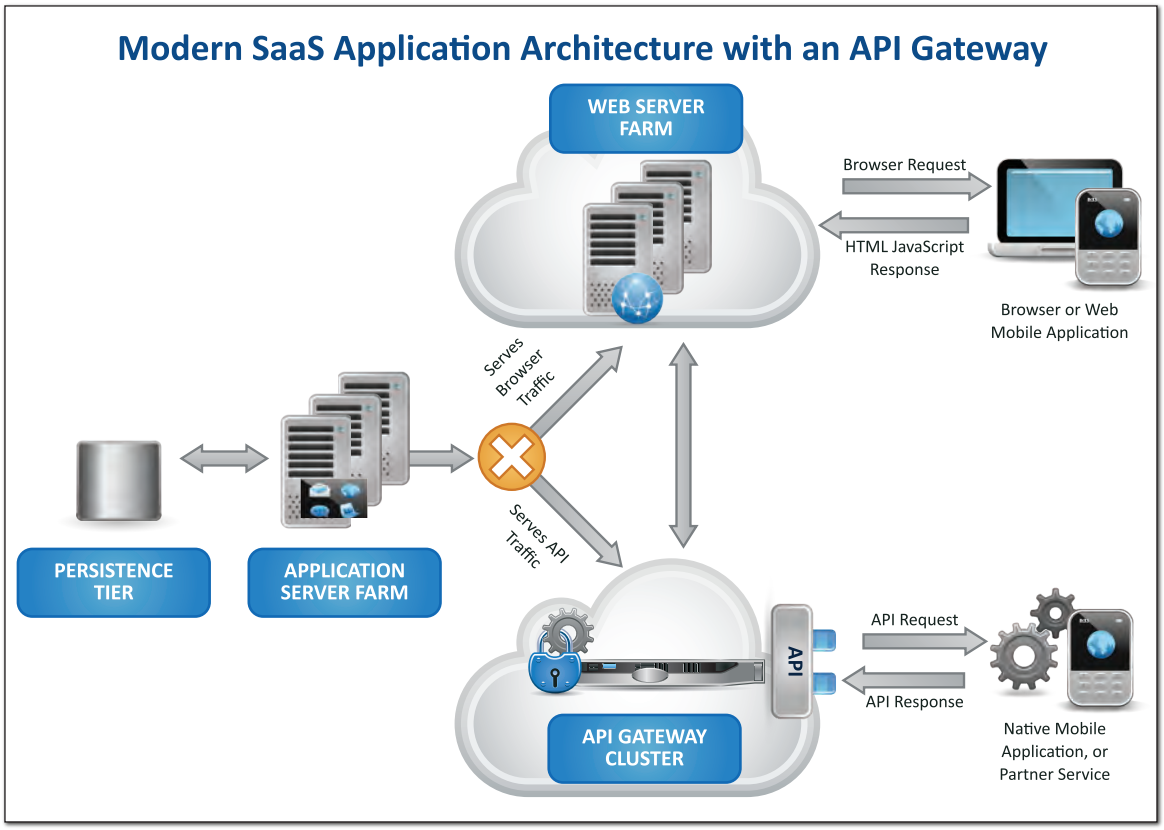

Figure 10: SaaS Pattern

CITO Research
Advancing the craft of technology leadership

Monetizing an API

Business usage context: Some enterprises generate revenue by "monetizing" assets that did not formerly carry a charge or whose characteristics change in the app economy. This might include extending an existing service to a partner with which the enterprise plans to share revenue, charging customers per-use, or preventing third parties from counterfeiting a product. For example, the telecom industry is moving toward monetizing APIs. Right now, everyone pays a flat fee for voice and data services. In the future, telcos will use API service calls as the basis for charging customers. API monetization models are becoming a centerpiece for many vertical industries and an API gateway delivers a tremendous "first mover-ready solution" advantage.

API gateway solution: An API gateway can provide tracking, metering, auditing, and throttling for API-delivered services, helping enterprises get fine-grained analytics and meet service-level agreements (SLAs). API gateways can track and meter user activity so subscribers can be charged for content they consume. This can also power analytics on customer behavior and improve future offers.

API gateways can audit activity to ensure proper use. For example, a company running a news site may want to prevent copying and repurposing news content. They can throttle data once certain thresholds have been crossed, protecting operational integrity. API gateways can also meter activity to support SLAs & prioritize traffic. If you want to know how well your API is working, the gateway can monitor the success rates of transactions, amount of uptime, and so on.

This paper was created by CITO Research and sponsored by Intel & McAfee Cloud Security Platform

Sponsor: Intel & McAfee Cloud Security Platform

Intel & McAfee deliver a portfolio of client, network endpoint, and data center cloud security products that are connected by a model of identity aware security. As users, data, and application services are exposed and consumed to and from the cloud; we establish a layer of trust that spans hardware, software, and cloud traffic channels. Technology enablement platforms include the McAfee Cloud Security Platform (Identity/SSO, API Security, email, threat protection, DLP), Nordic Edge OTP (mobile strong authentication), Intel® Trusted Execution Technology (Data Center Root of Trust), and Intel® Identity Protection Technology (hardware assisted client authentication).

About Intel®Expressway Service Gateway (Intel®ESG)

The API Gateway design pattern outlined in this paper can be implemented by deploying Intel®ESG as a software appliance, virtualized appliance in the cloud, or as a tamper resistant hardware appliance form factor with Common Criteria EAL L4+ certifications. Also, available as part of the McAfee Cloud Security Platform as McAfee Service Gateway. Intel®ESG is a mature product offering that has been deployed widely by fortune 100 corporations, federal governments, and cloud providers to protect SOA, Cloud, and Hybrid service implementations. Intel®ESG has been ranked a "top developer focused product" by leading analysts and offers true developer level customization & flexibility. For more, visit: *cloudsecurity.intel.com* or *mcafee.com/cloudsecurity* for video white board tutorials, cloud powered trial evals, and our webinar series on API Management.

CITO Research
Advancing the craft of technology leadership

CITO Research

CITO Research is a source of news, analysis, research, and knowledge for CIOs, CTOs, and other IT and business professionals. CITO Research engages in a dialogue with its audience to capture technology trends that are harvested, analyzed, and communicated in a sophisticated way to help practitioners solve difficult business problems.

About The Contributors

Dan Woods, Chief Analyst, CITO Research

Dan has written or coauthored more than 20 books about business and technology, ranging from books about service-oriented architecture, open source, manufacturing, RFID, and wikis to the ideas driving the latest generation of enterprise applications, particularly in the face of Web 2.0's impact on the enterprise. Dan's most recent book is *APIs: A Strategy Guide*. Since July 2008, Dan has been writing a column for Forbes.com.

Andy Thurai, Chief Architect & CTO, Application Security and Identity Products, Intel

Andy Thurai is Chief Architect and CTO of Application Security and Identity Products with Intel, where he is responsible for architecting SOA, Cloud, Governance, Security, and Identity solutions for their major corporate customers. In his role he is responsible for helping Intel/McAfee field sales, technical teams and customer executives. Prior to this role he has held technology architecture leadership and executive positions with L-1 Identity Solutions, IBM (Datapower), BMC, CSC, and Nortel. His interests and expertise include Cloud, SOA, identity management, security, governance, and SaaS. He holds a degree in Electrical and Electronics engineering and has over 20+ years of IT experience. He blogs regularly at http://www.thurai.net/securityblog on Security, SOA, Identity, Governance and Cloud topics.

Blake Dournaee, Product Manager, Intel Application Security and Identity Products

Blake is currently the product manager responsible for Intel© Expressway Service Gateway and an expert on API security. Blake was a specialist in applied cryptography applications at RSA Security. Blake is an established author who wrote the first book on XML Security and co-authored *SOA Demystified* (Intel Press).

John Musser, Founder, ProgrammableWeb

John Musser is the founder of ProgrammableWeb.com, the leading online resource for mashups, APIs, and the web as platform. During his 20-year career in software development he has shipped six award-winning software products in three industries working with companies including Electronic Arts, Credit Suisse, MTV, and Bell Labs. He has taught at Columbia University and the University of Washington and has written for a variety of technology publications on software development, most recently as author of *Web 2.0: Principles and Best Practices* (O'Reilly Media).