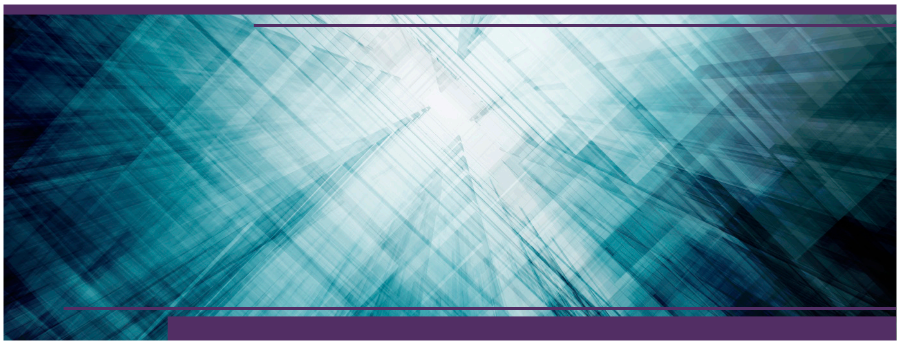
Understanding Your Enterprise API Requirements

Part 2: The 3 API management platforms – which architecture model fits your business?



Strategically choosing the right API management architecture model will ensure your APIs have the security, integration capabilities and performance necessary to deliver results.

As application programming interfaces (APIs) become integral to your business, managing those APIs from the right technology platform can help ensure they have the security, integration and performance capabilities you need.

As you learned in Part 1 of this series, API management platform architectures are driven by three main factors: the type of APIs you need to deliver, the readiness of your source APIs, and the nature of your integration requirements. This information, combined with what you learn in this paper, will help you choose the right API management architecture model for your organization.

If your requirements are basic, a commercial API management portal may be sufficient, and there are many on the market to choose from. If your requirements are more complex, however, you will need a more robust and flexible API management platform that offers a far higher level of security, integrity and integration capabilities.

This paper compares three incrementally more powerful API management reference architectures to suit a range of requirements and organizations. The three models are:

- All-in-one API management portal for simple delivery of consumer APIs
- Two-tier API delivery platform for scalable delivery of enterprise APIs
- Three-tier API switching network for intermediation of a large number of service-provider APIs

The all-in-one API management portal is suitable for delivery of consumer APIs with no strong integration, security or compliance requirements.

1. The All-in-One API Management Portal

The all-in-one portal is suitable for delivery of consumer APIs that do not have intensive integration, security or compliance requirements. The all-in-one option puts strong emphasis on the distribution phase of the API lifecycle, and its architecture is based on a developer portal. This type of portal consists of the following integrated components:

- Configurable portal
- Documentation delivery tools such as a wiki or other more advanced interactive tools
- Community forum for community building and peer support
- Self-service developer on-boarding and administration
- Self-service registration and configuration
- Lightweight proxy, throttling and routing capabilities
- Lightweight developer tools
- User, key and service repositories
- Business-centric monitoring and reporting

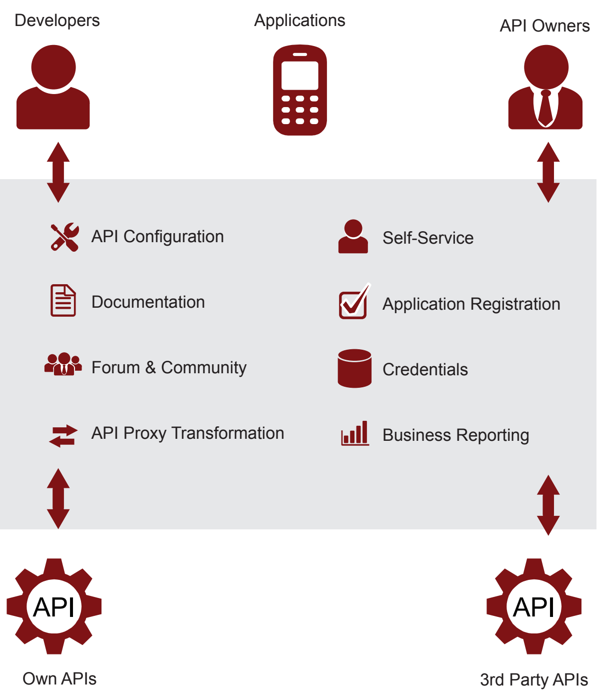


Figure 1: All-in-one API portal

The all-in-one model provides a greater level of simplicity than two-or three-tier platforms, but it does so at the expense of flexibility. Integration with enterprise systems typically requires custom code and professional services.

The all-in-one portal can be deployed locally, although it is usually offered as a cloud-based service. Cloud-based deployment can enable additional API exchange and distribution communities across multiple companies. However, depending on your enterprise information security policies, the multitenancy of cloud-based portal offerings may be too limiting for your organization. The all-in-one portal is appropriate if:

- You are delivering consumer APIs only
- Your backend APIs are ready-to-use without significant development for transformation
- The identity repository can be a stand-alone silo
- Security and identity management integration across API consumers, third-party API providers, and your own IT is not required
- No evidential audit trail is required
- There are few APIs
- There are no compliance requirements such as PCI DSS or HIPAA involved with the APIs
- There are little to no traffic-management requirements
- Little to no metering, billing or audit support is needed
- Little to no internal governance is required
- Minimal support is required from enterprise IT

2. The Two-Tier API Delivery Platform

A two-tier API delivery platform provides a more flexible and scalable solution for API management by separating the portal tier from an additional API gateway tier. The API gateway tier provides full API lifecycle support by integrating with internal and external systems such as identity management, partner management, and service registry systems. The API gateway tier provides the runtime engine for API delivery, as well as a single set of infrastructure services to support multiple portals.

A two-tier API delivery platform with separate portal and gateway tiers is suitable for delivery of enterprise APIs.

Portal Tier

The developer portal tier can be custom built or commercial. It may contain multiple developer portals, each designed for a specific subset of API consumers, such as your employees, business partners, service providers or the open community. Each portal is optimized with its own subset of enablement capabilities for a specific targeted API consumer community. Key components of the portal tier include:

- A configurable or custom-designed portal
- Documentation delivery tools such as a wiki or more advanced interactive tools
- A community forum for community building and peer support
- Self-service developer user interface for on-boarding and administration
- Self-service user interface for application registration and API key issuance

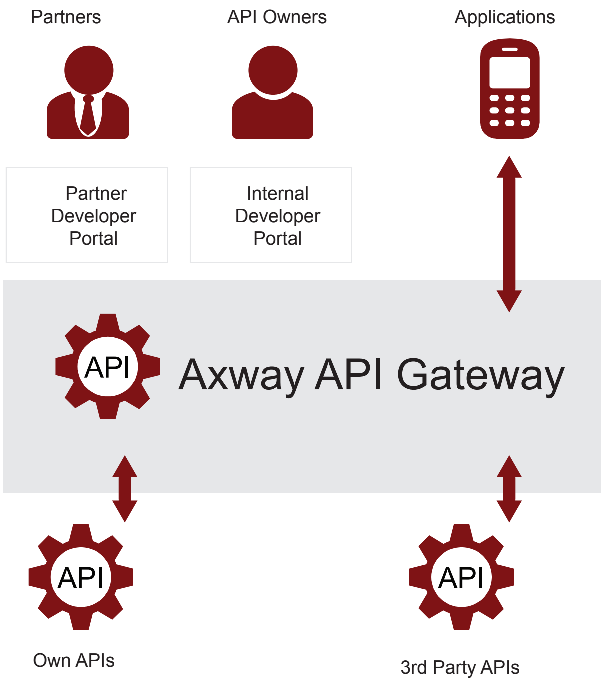


Figure 2: Two-tier API delivery platform

Developer portals for employees have the most advanced capabilities, including integration with:

- Portfolio management and SOA governance tools
- Source-code control and bug-tracking systems
- System management reports and alerts
- Resource request and provisioning systems
- Workflows and approval processes
- Security audit, risk assessment, and exception-lifecycle management processes

You can support multiple developer portals from a single API gateway infrastructure.

API Gateway Tier

The API gateway tier, usually a commercial offering, provides infrastructure services for developer portals and runtime for API delivery. Commercial gateways offer:

- Out-of-the-box integrations for a broad range of third-party systems
- Secure deployment at the edge of the enterprise
- High performance and ability to handle large traffic volume
- Rich administration capabilities for daily operational needs

Building a custom API gateway layer using products such as an enterprise service bus can result in a fragile, low-performance, and unsecured API delivery platform. Key capabilities to look for in an API gateway include:

- Traffic management
  - Request and response routing
  - Throttling and metering
  - Quota management
  - Service-level agreement (SLA) management
  - Caching
- Security
  - API key and certificate management
  - OAuth and SAML federation and trust relationship management
  - Authentication and single sign-on (SSO) capability
  - Strong, multi-factor, multi-tier authentication
  - Coarse-grained, fine-grained and contextual authorization
  - Encryption, decryption and tokenization
  - Configurable audit and logging
  - Content firewalling
- API mediation
  - API virtualization
  - Protocol translation (e.g. SOAP/XML to REST/JSON and vice versa)
  - Data and payload transformation
  - Data redaction
  - Message enrichment
- Monitoring and reporting
  - Transaction logging, tracing and debugging
  - Service statistics reporting
  - SLA monitoring and alerting
  - Real-time monitoring
- Portal enablement services
  - Organization, application, and user lifecycle management
  - User enablement and disablement
  - Service lifecycle management
  - API key generation, renew and re-issue

- Security integration tools
- Usage reports
- Billing reports
- Service-level reporting

Integration is a key driver for adopting a two-tiered architecture. Instead of creating custom integrations that are difficult to develop and expensive to maintain, an API gateway tier provides out-of-the-box, configuration-driven integrations. The following diagram illustrates common integrations:

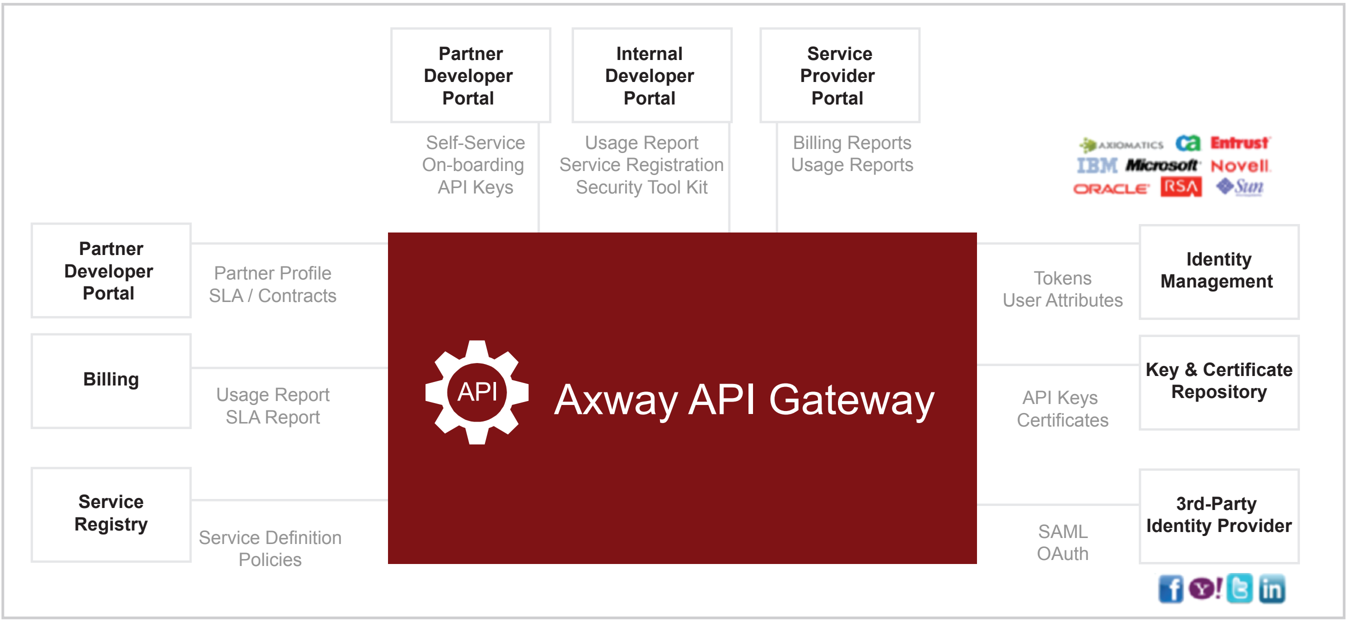

Partner Developer Portal
Internal Developer Portal
Service Provider Portal
Self-Service On-boarding API Keys
Usage Report Service Registration Security Tool Kit
Billing Reports Usage Reports
Partner Developer Portal
Partner Profile SLA / Contracts
Billing
Usage Report SLA Report
Service Registry
Service Definition Policies
API
Axway API Gateway
Tokens User Attributes
Identity Management
API Keys Certificates
Key & Certificate Repository
SAML OAuth
3rd-Party Identity Provider

Figure 3: Typical API gateway integrations

An API gateway provides out-of-the-box integrations with leading identity management platforms.

When to Use a Two-tier API Delivery Platform

A two-tier architecture can be deployed locally, in the cloud, or in a hybrid model. Depending on the extent of the integrations, an on-premise API gateway may be required. Portals can be deployed either on-premise or in the cloud, independent of the API gateway deployment location, and regardless of the deployment location of other portals. Note, however, that while there are compelling reasons to deploy partner and provider portals in the cloud, internal developer portals are best suited to on-site deployment.

Use the two-tier API delivery platform option if:

- You have enterprise APIs
- You have backend APIs that require non-trivial transformation and orchestration
- A standalone identity repository for API management is not acceptable
- You have non-trivial access federation scenarios

- You need support for existing trusted relationships, security protocols and certificates
- Integration to existing systems across the network boundary is difficult
- You need integration with partners or your own identity-management platforms
- You have more than a handful of APIs
- You need to integrate with a service registry or SOA repository
- Your APIs are subject to compliance requirements such as PCI DSS or HIPAA
- You have non-trivial traffic-management requirements such as quota-based throttling, dynamic routing, and contextual load balancing
- You have configurable metering and logging or support billing and audit requirements
- You have APIs for long-running services
- You have contract-binding SLAs
- You need content firewalling

3. The Three-Tier API-Switching Network

If your business plays an intermediary role and acts as the broker between a network of service providers and channels, a three-tier API switching network will provide you with the most scalable and flexible API management platform. The API switching network architecture adds an API broker layer to interface with outbound service-provider APIs. The API broker serves two purposes:

1. It isolates the API gateway and any intermediary business-logic components (e.g. Business Process Management or an Enterprise Service Bus) from the complexity and change introduced by service providers
2. It provides additional flexibility to enrich source APIs

The API gateway makes security and traffic management decisions based on partner profiles, SLAs and contracts.

An API gateway enables secure external delivery of any existing SOA, B2B and legacy application interfaces.

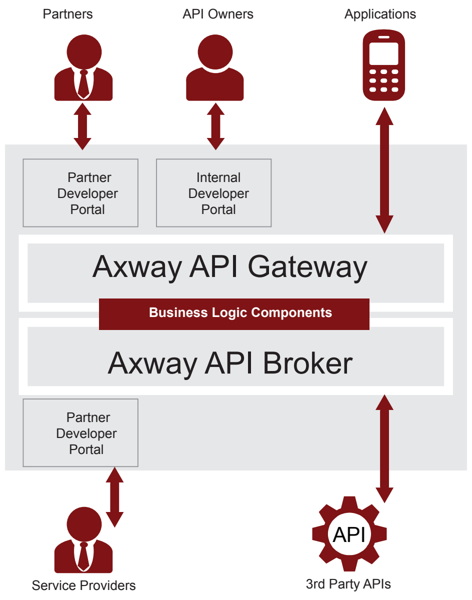


Figure 4: Three-tier API switching network

Using an API Broker for Isolation

Due to the sheer volume of service providers and the accelerating pace of business today, the rate at which service provider APIs change usually outpaces that of the internal API management platform.

For example, let's say your API switching network has 100 service providers, each with three APIs, and each API goes through four revisions per year. That equates to 1,200 changes per year your API management platform will need to accommodate. Some of these changes may include new functionalities or bug fixes that you want to incorporate and propagate across your API management platform. However, most of the revisions will not seem important enough to warrant disturbing your platform. While you can make the choice to opt out of those changes, and refrain from using new versions of APIs, this is likely to have long-term effects and can often lead to support issues down the road.

An API broker allows you to create virtual APIs to buffer these external changes, so you don't have to opt out.

As another example, let's say you are providing a service to view credit reports and you have credit bureaus A, B, and C as service providers. Each credit bureau provides you its own API to pull individual credit reports and each uses a different type of authentication token for its API. Instead of exposing the three sets of APIs directly to your internal and channel developers, you can use the API broker to create one virtual API used to extract credit reports from any of the credit bureaus. This makes it extremely simple for developers to adopt your API for credit reporting.

An API switching network enables powerful digital commerce intermediary business models.

The API broker also helps you to manage the three different authentication tokens. The tokens are safely stored and managed in the API broker, and the broker dynamically inserts the appropriate token depending on which credit bureau's API is being invoked. The API broker not only improves ease-of-use of provider APIs, but also enhances the security posture of the API management platform.

To continue with this example, say bureau A has completed a minor update to its API by adding a new query parameter that does not affect your business, but it does introduce a minor variation to the formatting of the query string. Instead of propagating this change through to your API consumers, you can simply buffer the change at the API broker. Existing API consumers experience no change at your virtual API, but when the API broker calls the new credit bureau A API, it will make the adjustment and use the new query syntax.

In another scenario, say your business has decided to terminate its relationship with credit bureau A and starts sourcing from credit bureau D. With the virtual API deployed on the API broker, the available list of credit bureaus for the credit reporting API is stored as a list of external parameters. To terminate credit bureau A and add credit bureau D is a simple task of updating the list of approved credit bureaus and configuring the new credit bureau D API in the API broker. The API management platform users see that the list of available bureau choices has changed, and that no change has been made to the underlying credit reporting API.

Using an API broker to isolate external changes in backend service-provider APIs can simplify API adoption, reduce the number of changes, and improve the overall quality of the API switching network.

Using an API Broker for Mash-up

The new generation of applications is all about creating compelling capabilities by integrating existing capabilities in interesting ways. You can for instance, create an application to find local restaurants by combining APIs that provide: a database of local businesses, restaurant reviews, Twitter posts, mobile phone location services, and Google Maps. This is an example of a mash-up API.

The API broker not only provides the ability to create virtual APIs for isolation, it also enables you to create mash-up APIs. With an API broker you can:

An API broker manages and mediates security tokens across service-provider APIs.

- Create APIs that aggregate data from similar services, such as combining product reviews from multiple review websites
- Create APIs to deliver integrated functions, such as combining product reviews, low-price search and a retail store location map
- Create APIs to enrich the user experience, such as letting them scan a bar code with their mobile phone camera to see a list of how-to videos on how to use a product

An API broker enables you to create APIs that are better than what your partners provide out-of-the-box; improves overall ease of use; and allows you to create your own differentiated services.

Features of the API Broker

API broker creates differentiated mash-up APIs from service-provider source APIs.

Similar to the API gateway, your API broker will likely be a commercial product. Commercial brokers offer capabilities that are difficult to create in a custom-built platform, such as:

- Out-of-the-box integrations for a broad range of third-party systems
- Secure deployment at the edge of the enterprise
- High performance and ability to handle large traffic volume
- Rich administration capabilities for daily operational needs

API broker technology goes by many different names in the marketplace, including API gateway, API broker, cloud services broker, and cloud gateway. Look for these key capabilities when shopping:

- Traffic management
    - Request and response routing
    - Throttling and metering
    - SLA testing and auditing
    - Caching
- Security
    - API key and certificate management
    - OAuth and SAML federation and trust relationship management
    - Token caching, single sign-on and impersonation
    - Strong, multi-factor, multi-tier authentication
    - Encryption, decryption and tokenization
    - Configurable audit and logging
    - Data leakage monitoring
- API mediation
    - API virtualization and aggregation
    - Orchestration
    - Protocol translation (e.g. SOAP/XML to REST/JSON and vice versa)
    - Data and payload transformation
    - Data redaction
    - Message enrichment
- Monitoring and reporting
    - Transaction logging, tracing and debugging
    - Service usage metering and reporting
    - SLA monitoring, audit and alerting

Note that integration is as important to the API broker as it is to the API gateway. Both layers require essentially the same set of integration capabilities.

Choosing a Three-Tier API Switching Network

Your three-tier architecture can be deployed locally, in the cloud, or as a hybrid, depending on your integration requirements.

The three-tier architecture can be deployed locally, in the cloud, or in a hybrid model. Depending again on the extent of your required integrations, the API gateway and broker components may need to be on-premise. Portals can be deployed either on-premise or in the cloud, independent of the API gateway and broker deployment locations.

In some cases, it may be best to deploy API brokers both in the cloud and on-premise as a trusted-pair. The on-premise broker can treat virtual APIs from the cloud-based broker as a set of trusted cloud-based services. The trusted-pair deployment pattern can also have the remote broker deployed on-site at a service provider. This deployment architecture is especially useful when a service provider has integration requirements that are difficult to achieve without an on-premise footprint.

Use a three-tier API switching network if:

- You have enterprise or consumer APIs
- You have APIs from a large network of service providers
- You have an intermediary business model that requires abstraction of a service-provider API from service consumers, and vice versa
- You have multiple vendors providing the same service with different APIs
- There is a high frequency of changes to your service-provider APIs
- You have an integrated API offering that combines capabilities from multiple service-provider APIs
- Your backend APIs require non-trivial transformation and orchestration
- You have a variety of identity-management and security integration requirements across many service providers
- You require assistance with integration via a broker deployed on-site at the service provider

The Importance of Architecture

Choosing the right API management platform architecture is key to attaining the level of security, integration capabilities and flexibility you need for a successful API program. First understand the type of APIs you need to deliver and consume, assess the requirements of integration with existing infrastructure, then decide on one of the three reference architectures presented in this paper. Once you have a sound API management strategy you can begin putting it into action.

The time is now. Make the technology of the API go to work for your customers, your partners, your communities – and your bottom line.

For More Information, visit www.axway.com