The business of APIs: Best practices

Contents

Executive summary

Many enterprises are planning their journey and participation in the API economy. Some view this plan as an IT initiative, but those who truly understand the potential are attacking it with a combination of business and IT perspectives. This paper focuses on best practices to drive success using this blend of business and IT.

Determining an API economy strategy and planning a roadmap have several significant benefits:

- Consolidating and standardizing common APIs—or simply business services—within an organization
- Lowering cost of operations by having a central repository and index of enterprise business services such as retrieve credit score
- Accelerating digital projects and improving time to market with safe, quick access to business services by both internal and external parties
- Identifying a partnership ecosystem—especially outside your own industry—to formulate new value-add products and services to be more competitive
- Defining a new business model for monetization purposes such as a mobile marketplace; that is, curating your company's business capabilities aggregated with your partners' business capabilities to provide a diverse range of related or complementary services

In this paper, we share best practices and lessons learned based on interactions between IBM and its clients. The paper focuses on the core nontechnical aspects of executing an API initiative, including:

- Business strategy
- Domain ownership and organizational structure
- Governance model
- Monetization
- API identification
- Communication
- Legal and privacy
- Success criteria and metrics
- Technical governance

For each area, we describe the issues and offer recommendations. In addition, we provide sample worksheets to help you organize your approach within that segment of the initiative.

This paper is intended for business and IT leadership interested in jump-starting their API initiative by learning about best practices being used by other enterprises. It is not meant to be introductory, and prior knowledge of the basics of business APIs and the API economy is expected.

The business of APIs: Introduction

Emerging entrants and companies initiating new, market-changing offerings are disrupting many industries. Here are just a few examples:

- Apple Pay is disrupting payment services
- Smart cars are disrupting the automotive industry
- Online commerce enterprises such as Alibaba are investing in the property and casualty insurance industry, disrupting it with new service models and offerings
- Financial technology companies (fintechs) such as Betterment and Bats Chi-X have disrupted the high barrier to entering financial markets
- Uber has disrupted transportation
- Netflix has disrupted the media and entertainment industry

APIs give companies a mechanism to become the disrupter, or at a minimum to more rapidly respond when a disrupter enters their market segment. They offer an opportunity to bring significant value to a business. To capture this value, business and IT need to work together on the API initiative. This paper is intended to address the less-technical—but no less challenging—issues of an API initiative:

- Business strategy
- Domain ownership and organizational structure
- Governance model
- Monetization
- API identification
- Communication
- Legal and privacy
- Success criteria and metrics
- Technical governance

Often, the early phases of an API initiative are led by IT, which in turn is focused on the technical implementation, architecture and security implications of adding APIs to the current IT environment. All of these topics are important but are not covered in this paper. Visit the IBM® API Connect™ site on IBM developerWorks® and the IBM API Connect product website to learn how IBM can assist you with these technical aspects.

Additionally, this paper is targeted toward enterprises that see APIs as a platform strategy, not an individual project. Individual project orientation may result in a quick initial project, but will not provide the best practices to drive the repeatability required to move forward at the enterprise level. Many of the most successful businesses using APIs view them instead as a corporate channel to market—a strategic asset.

Throughout the paper, we include sample worksheets to help you gather and organize your approach to each topic. Some of these worksheets have fields prefilled with suggestions. You can use these worksheets to organize the initiative with your own data and choose to use or disregard the samples shown.

Business strategy

Why are you planning to create APIs? If you cannot answer this question at a business level, please stop the initiative and regroup.

Some IT divisions within organizations begin API implementations without clear-cut business use cases. These initiatives will find success difficult because there is no defined business goal or goals. Those goals may be driven by revenue, new routes to market, new value-add products and services, efficiencies, time to market or other elements, but they must be outlined at the start so all decisions and actions can progress toward the goal.

Companies that are executing successful API initiatives focus on one or more of these four key business drivers:

- **Speed (also known as two-speed IT, bimodal IT or multispeed IT):** Typically the first driver of API use in an enterprise, this element is focused on allowing business and IT organizations to run at different speeds. Traditional IT management of core systems of record can be changed at a certain rate. Trying to force rapid changes into these core enterprise systems could result in outages or security exposures. Yet the business needs to react very quickly to new opportunities and competitive threats. It needs a higher rate of change than can be delivered by the controlled changes required by the systems of record. Using APIs, you can prepackage core system assets for consumption by the business to create new and innovative systems of engagement.
- **Reach:** To reach new markets and obtain new customers, you can make APIs available to other enterprises—for example, partners—who through their interaction with clients will generate additional revenue and customers for your enterprise. For example, an insurer may work with a low-cost smartphone manufacturer to preinstall its insurance app and enable opt-in push notifications for relevant location-based information and interaction. The goal is to gain access to younger generations of users in over 20 emerging growth markets.
- **Internet of Things (IoT) or devices:** In many industries, devices are used in conjunction with APIs to provide new and innovative solutions. This tends to happen in one of three ways:
  1. A device sends data through an API call, such as a connected car sending data on driving patterns to an insurance company.
  2. A device is sent a command through an API call, such as a security office issuing a command to pan a remote security camera to the right.
  3. A device sends data through a non-API call using other technology such as MQTT—a high-volume messaging protocol and transport for telemetry devices—because all the data does not require action. However, APIs are used to access the data inside the enterprise and look for or react to particular situations or events. For example, medical monitoring devices are constantly sending data and analytics are used on the data to try and spot problems. If a problem is found, APIs are used to alert the doctor and patient.
- **Domains:** Domains typically refer to interactions across multiple lines of business. They can largely work independently, but benefit from sharing data. APIs allow the data to be shared in a controlled, secured manner. Domains can also be seen as physical locations. Companies that have multiple locations, which may include cloud and on-premises data centers, sometimes use APIs as a method to secure and control the flow of data between locations.

Business goal	Priority	Time frame	Internal sponsor	API audience
Financial				
Market share				
Internal development				
Partnering				
Competition				
Time to market				
Innovation				
Regulatory				

Figure 1. Business strategy worksheet.

There are plenty of reasons why your business could or should be interested in an API initiative; beyond the four listed previously, other common drivers include:

- Mobile development and internal use
- Partnering and partner onboarding
- Financial; API monetization
- Time to market
- Competitive pressure
- Regulatory requirements
- Innovation

Whether on this list or not, the reason for the API initiative should be clearly understood. It is common to have multiple reasons for using APIs, as is having a prioritized multistage view of when each target area will be addressed.

Another aspect of business strategy is defining the audience for the APIs. We typically think of three potential audiences: internal employees, partners and public consumers. However, you could define further breakdowns for different internal audiences—such as lines of business, types of partners, suppliers versus distributors and so on.

A worksheet can help you document goals and define your audience when laying out your strategy. A sample worksheet is shown in Figure 1.

Domain ownership and organizational structure

Almost every company has multiple lines of business that can benefit from exposing assets as APIs, in addition to the central IT organization that will be involved in the initiative. Effective API efforts need to address how these organizations will work together.

A common best practice is to form a group in the IT organization to own the API technical implementation and infrastructure. Whether the deployment is on premises or in the cloud, a central team providing consistent methodology and tooling across the enterprise will result in cost savings and reduce duplicate efforts. Agreement on tooling platform selection and skills enablement will come into play because the partitioning of control and funding are renegotiated across the traditional organization lines (by matrix or otherwise) in an API situation.

The IT organization will establish the environments (development, test, QA, production and so on) in which to implement the tooling and infrastructure to help with API creation, runtime, management and security. These tools can also be used to develop the methodology for deploying APIs into the environments and, if they are on premises, to implement the operations of the environment. Interoperability of technology and people should be a focus area regardless of ownership domains. However, this team will not define the business APIs.

Enterprises that have already undertaken a service-oriented architecture (SOA) initiative are in an excellent position to move forward with an API initiative. SOA initiatives provide a robust platform to build upon. And the services identified in the SOA are the foundation that will be used for the APIs. Marrying the API initiative and SOA initiative, and understanding when a requirement should drive API creation and when it should drive a service, is a critical set of criteria that needs to be defined by the core API team.

If an enterprise has not previously implemented an SOA infrastructure, should it do so first? The answer is no. Waiting to build an SOA infrastructure would result in extreme delay and potential lost business opportunities that would have been enabled through APIs. A better approach is to implement your API initiative first, and then let this help drive the creation of your system-of-record SOA services.

Who should drive the API initiative?

One of the most common mistakes that businesses make is assigning the IT team that built their SOA services and infrastructure the task of executing the API initiative by themselves. While the work these teams did may have been excellent, the purpose of the API initiative is different than that of the SOA. If the group creates APIs as an interface to each service that was built for the SOA, they will just add another layer—missing out on all of the potential value that could be obtained by the API. While some APIs may call one existing service, this is not always going to be the case. The team needs to take a business-oriented or consumer-oriented approach. We will discuss this point further in the “API identification” section.

Many businesses struggle with the boundary between APIs and SOA services. No hard and fast rule is available. Some best practices in this situation are to not think about technology, but rather the purpose of the asset being created. If the asset is an update or new core business function that will be useful to all consumers and needs to be thoroughly planned and governed, it is an SOA service. If the asset is about making a core business function consumable quickly, enforcing security and other policies, and meeting the needs of a consumer in a tailored fashion around the core systems, then it is an API. Having a view across the complete solution—consumer to API to service—provides the best results for each component.

So, who should identify the business APIs?

Figure 2 highlights many roles in a high-level organizational structure. Note that several people may be in each role and a single person could be assigned to multiple roles.

A critical role in the structure is the API product manager, which is typically a new role in most enterprises. The person or people in this role own the success of the API or APIs and the API initiative. Therefore, this role requires strong leadership skills, as API product managers will be working across departments and must influence parts of the organization they do not control. While primarily business-oriented, the API product manager needs to bridge to the technology side.

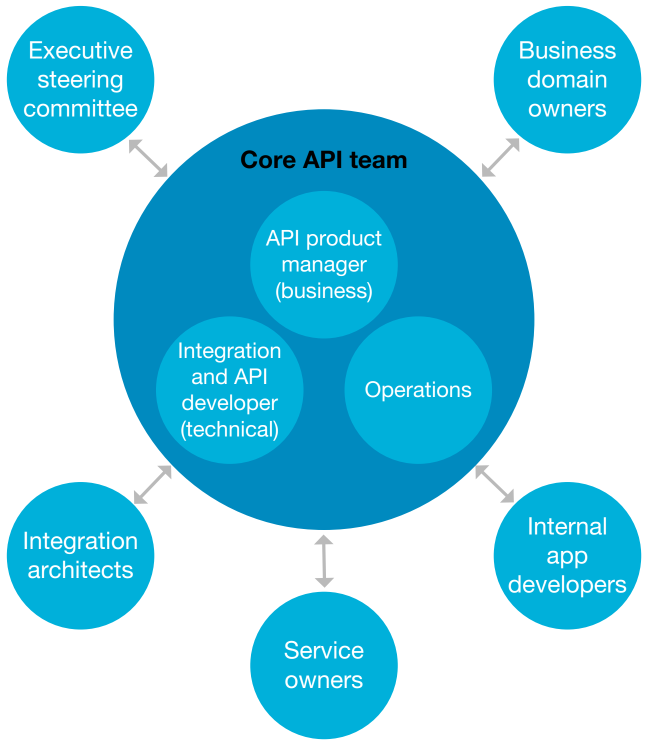


Figure 2. High-level organizational structure for an API development team.

Role	Person(s) assigned
API product manager and initiative leader	
Business leader (if different)	
Technical leader	
Executive steering committee	
Governance leader	
API and domain owner 1	
API and domain owner 2	
API evangelist	
Marketing and communication plan owner	
Integration architects	
App development leader (internal)	
Partner interface	
Test leader	
Legal interface	
Finance interface	

Figure 3. Role assignments worksheet.

Key tasks associated with the API product manager role include:

- Working with the domain owners to identify desired business APIs to bring to market
- Working with the API developer to drive creation of the API
- Reporting to executives on metrics
- Defining the product characteristics of the API (monetization, rate limits, audience and so on)
- Communication

The other roles depicted in the organizational structure, shown in Figure 3, already exist or have similar roles in most enterprises:

- **API developer:** This technical role creates, tests and deploys the APIs. The skill set is similar to integration specialists.
- **Operations:** This traditional operations role ensures availability and service-level agreements (SLAs). If deployed in an off-premises cloud such as the IBM Bluemix® platform, then the cloud provider is acting in this role.

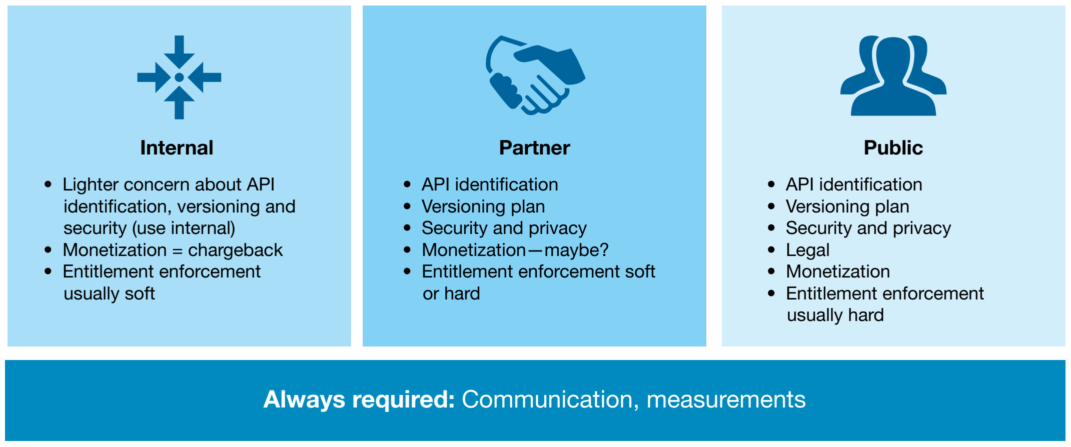

Internal
• Lighter concern about API identification, versioning and security (use internal)
• Monetization = chargeback
• Entitlement enforcement usually soft
Partner
• API identification
• Versioning plan
• Security and privacy
• Monetization—maybe?
• Entitlement enforcement soft or hard
Public
• API identification
• Versioning plan
• Security and privacy
• Legal
• Monetization
• Entitlement enforcement usually hard
Always required: Communication, measurements

Figure 4. API governance growth by audience.

- **Domain owners:** These organizations own the business assets. Ensuring these groups are involved in the API identification process, have bought into the initiative and understand the value proposition is critical.
- **Executive steering committee:** This committee provides executive, funding and resource commitments. The API product manager needs to provide them with measurements and reports.
- **Integration architects and service owners:** These traditional roles represent the existing IT services.
- **Internal application developers:** These groups are the internal target audiences for the APIs. There may be many of them, such as the mobile app development team or a different line of business (LOB) consuming APIs supplied by the original LOB.

Governance model

Just hearing the word "governance" makes many cringe. The challenge for API governance is to have the correct amount of governance without it becoming a roadblock to innovation. One of the key drivers for an API initiative is speed, so lengthy governance processes could defeat the entire initiative. Typically, systems of record (and the SOA services that represent them) require robust governance processes for the availability of these systems. Assets exposed through APIs should not have these same governance requirements, thus allowing for a more expedited, lighter-weight governance process.

Many businesses take a staged approach to API audiences by starting with internal consumers, moving to partners later and potentially moving to public APIs in the future. As the initiative grows and audiences stretch further from your organization's control, the governance processes can likewise mature and grow. The best practice is to start with a small governance process and add as necessary. Figure 4 shows a potential growth path for governance.

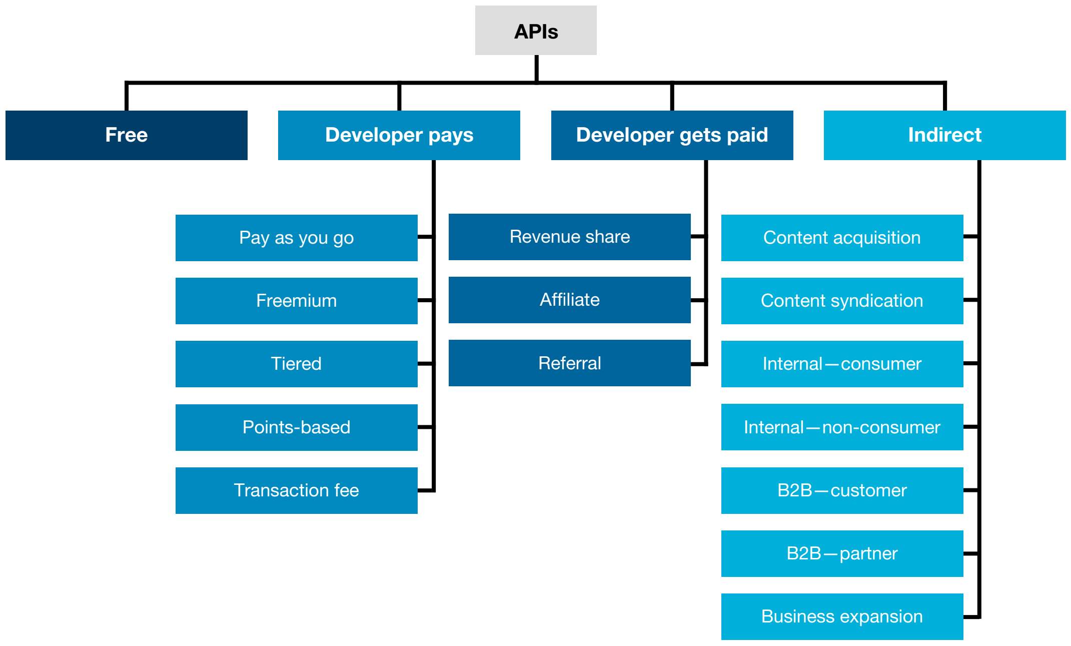


Figure 5. API monetization business models.

Information governance is an especially important best practice when industry standards are involved, such as in finance and healthcare. API development can proliferate through enterprise innovation, but the payload data model and taxonomy should have some level of uniformity and predictability—more so if the business model is a back-end-as-a-service (BaaS) model or a business-to-business (B2B) channel expansion model.

Monetization

IBM published a separate paper on the monetization topic, titled “API Monetization—Understanding your Business Model Options.” In that paper, we focus on the business models for API monetization. We describe four groups of API models: free, developer pays, developer gets paid and indirect, with many sub-models for each. We explain each monetization model and provide examples of companies that are executing that model. Figure 5 shows a breakdown of the four primary models and their sub-models.

The paper also provides initiative guidance and considerations, plus a recommended project approach to implementing API monetization in your enterprise. Download the white paper here.

API identification

Identifying good APIs is one of the most critical factors in achieving API initiative (and associated business) success. APIs need to be focused on the needs of the consumer, and they should be simple. Three questions lead to a good API:

- Who is the audience?
- What do they want?
- Under what terms and conditions are you willing to make the asset available?

Notice that none of these questions mention the systems of record that will ultimately deliver the response to the API request. Many companies incorrectly define their APIs by looking at what the systems of record do and adding an API in front of them. This approach may simplify the process for the API provider, but it does not meet the needs of the consumer.

When identifying a candidate API, the API product manager needs to understand the target API user (question one). The second question is probably the most important of the three. Understanding what the audience is trying to accomplish can result in the best API interface. If the definition is focused on consumer need, then the interface is more likely to be useful to that audience and more likely to stand up to change (versioning).

If the interface to the API is not directly related to the back-end system of record resources, then the API does not necessarily have to change if the resources change their interface. A new backward-compatible API can be delivered and consumers can be migrated seamlessly to the new version. However, if the API is focused on the interfaces to the back-end system of record resources, then each time one of these systems changes, the API is likely to change as well. This non-backward-compatible version of the API can require consumers to update their application—which will be viewed poorly by the consumer if it happens often, and typically indicates to them that your APIs are not stable.

The third question is related to the policies you want to have around the API. What security measures are required to allow the API to be used correctly? Are there rate limits that must be enforced?

Once you have answered these three questions, the API product manager and API developer must work together and potentially iterate to define the API. The API developer needs to map the proposed consumer interface for the API to the back-end system of record interfaces and possibly to many other systems to provide only the desired result back to the consumer.

Working from the API consumer inward, new business logic may need to be added at a microservice layer in front of the existing systems of record. If the current systems do not completely address the requirement, additional coding may be necessary to add business logic to the existing environment. The ability to create and run new business logic in front of the systems of record allows for the required speed to deploy new business offerings quickly. Without this capability, the business will revert back to waiting for changes to the systems of record, which could elongate the initiatives' development cycles.

Many businesses have had tremendous success implementing a microservice approach based on creating and running new business logic in front of their core systems:

- GoDaddy is now serving hundreds of thousands of small businesses' websites, replacing legacy hosting platforms. The move to a microservice approach based on Node.js is improving customers' performance by 4 times and has reduced server infrastructure costs by 10 times.[1]
- Groupon refactored its approach by making parallel API requests to services. This approach resulted in improved site-wide page loads of 50 percent.[2]
- LinkedIn rebuilt its mobile app with a microservice (Node.js) back end. As a result, system speed improved 20 times while reducing the number of servers by 90 percent.[3]

You can use the worksheet shown in Figure 6 to aid your API identification efforts.

API	Use case(s)	Target audience(s)	Business model	Owning organization

Figure 6. API identification worksheet.

Communication

"If you build it, they will come" does not work in an API initiative. The best practice is to plan for and address the need for communication.

APIs need to be marketed to the target audiences. How will the audience know the APIs exist if you do not tell them? How will they learn about the benefits of using the API? Common approaches include executing "lunch and learn" sessions for internal audiences or using your partner channel communications to reach that audience. You can also publicize external public APIs on common sites such as www.programmableweb.com.

Running hackathons for your APIs is another idea. This type of event is a great method to communicate as well as obtain feedback and direction on your APIs. You may choose to attend or run events and conferences related to APIs. Doing so will help build your skills and also let you communicate and collect new ideas on how to get your message out.

Be sure to communicate the status of the initiative and the achievement toward the initiative goals to the executive steering committee. Communication helps the executive committee understand the initiative's value, which helps maintain the funding and drives future expansion of the initiative.

You may need to target communications to multiple, different audiences, so tailor the message to what each audience needs to know. Do not give the executives the same messages you are delivering to the app developers. Develop a communication plan to help determine the most important information for each audience. The worksheet in Figure 7 can help with this process.

Audience	Message(s)	Owner	Communication type(s)	Frequency
Executives				
Internal app developers				
Partners				
Public				
Business domain owners				

Figure 7. Communication plan worksheet.

Legal and privacy

Lawyers by nature are often cautious and may be nervous about making information available through this new API channel. This is similar to legal teams' concern and caution when the World Wide Web began rapidly expanding in use and pervasiveness. Use the web as an analogy with the legal team to explain that this channel is new and represents a significant business opportunity. Saying that the API cannot be done is not an acceptable response; the product manager will need to work with the legal team to find a satisfactory way to make it work. Be prepared to answer these questions to move past this hurdle:

- Do you own the data you are providing?
- Are the intended audiences entitled to access this data?
- What rights are you granting the consumer of the API to use the data provided?
- How will you communicate the terms of use to the API consumer?
- How are you ensuring privacy?
- What is the required policy for data retention?
- What requirements do you have for attribution of the content or use of your brand? Do you need to give attribution to some other entity?
- How will you find out and handle consumers who do not use the API appropriately?
- What are your liabilities?

Managing customer privacy is an important consideration. The appropriate level of security must be in place to ensure only authorized users access the customer's data. Apps need to be validated so they are allowed to access the API. The app user's identity must be secured so that the app itself does not have access to the user's credentials. OAuth is a common protocol used for this purpose. The customer's identity also needs to pass into the systems of record to ensure only that specific user's data is accessed.

Do not forget about organizational privacy as well. Several organizations that consume the same API may be competing with one another and should not be able to see each other's customers or data. Figure 8 provides a worksheet to help you determine the privacy and legal concerns involved for each API candidate.

API candidate	Privacy concern?	Legal concern?	Issue owner

Figure 8. Legal and privacy worksheet.

Success criteria and metrics

Establish meaningful, measurable goals for success and gain executive agreement up front. Common metrics include:

- App developer sign-ups
- API usage rates and rate of growth
- Number of apps driving usage of more than *n* transactions per time interval
- Revenue generated
- The type of data being requested, and the type of data not being requested
- Usage patterns, dates, locations—if any
- Whether app developers are using multiple APIs

Look at technical metrics as well to see where improvement is required:

- Are developers coming to the site and not signing up?
- Are API calls coming back with errors?
- Is performance acceptable?

Publish reports or make dashboards available to the appropriate audience for easy access to metrics. Figure 9 provides a worksheet to track details associated with measurements. You can customize the worksheet for the measurements you deem important for your audiences.

Measurement	Goal	Report type	Owner	Audience	Frequency
Developer registrations					
API usage rate					
Revenue					
Error rates					

Figure 9. Measurements worksheet.

Technical governance

As mentioned earlier, the focus of this paper is on nontechnical best practices. However, some borderline items that bridge both technical and nontechnical concerns are worth mentioning. Note that this list is not intended to be the complete view of technical governance. Here are a few recommendations:

- **API standards and best practices:** Representational State Transfer (REST) and Simple Object Access Protocol (SOAP) are the most common API technologies. Usually REST and JavaScript Object Notation (JSON) are used for lower-bandwidth devices and to reduce the amount of data transmitted. REST/JSON also tends to be simpler to understand and consume.
- **Naming standards:** Choose a naming standard that is meaningful to the organizations providing the APIs and easy for a consumer to understand.
- **Granularity and simplicity:**
  - APIs are fine-grained—not like SOA services—and they typically do one thing.
  - Easy-to-understand parameters: Aim for self-documenting parameters using easy-to-understand names. Supply samples to call the API and the returned data.
  - Don't do anything unusual. Unusual techniques will confuse app developers and make them less likely to use your API.
  - Make it easy to try your API.
  - The back-end systems behind the API interface may be complex, but do not show the complexity. One API call may access three back-end systems, extract lots of data, throw 90 percent of it away and reply back with the 10 percent that answers the API request. Do not supply three APIs and make consumers do the work of parsing each request's results to find what they need.
- **Lifecycle:** Require few stages. If there are too many stages, the cycle to make an API available will be too long.
- **Deployment and publishing:** Early environments, such as development and test, are usually handled synchronously. However, production is usually isolated and requires a disconnected deployment approach (usually scripted).
- **Versioning:** This recommendation is most critical for public APIs. Plan for change. Try to make APIs backward-compatible (as described previously), so you can move consumers automatically to the new level. Often a need exists for multiple versions in production at the same time. Make a plan to move consumers to the new versions and retire older versions.

- **Scale:** API entitlement levels are used to help plan for capacity, and the API gateway enforces these entitlements. With IBM API Connect, scaling should be simple using additional instances added to a cluster for the gateway or management clusters. The gateway can protect back-end systems of record from becoming overloaded. Watch the analytics to see how your requests are being managed. If your APIs are successful and you are generating more revenue, scaling the systems of record behind the API layer to handle the additional requests as well may eventually be necessary.
- **Integration:** Set up guidelines for integration and when something should be an API versus a back-end service. Where should the integration occur? Is a required new capability something that should be delivered as an API, or is the change really required in the systems of record? Watch out for inappropriate API proposals that are trying to take advantage of the lighter-weight governance around APIs.

Closing thoughts and recommendations

Do not wait until you know all the answers and have everything in place to get started with an API initiative. The market is moving too fast—an Uber, Netflix or Apple Pay could disrupt your space at any time. Plan stages for the rollout that build on what you learn and iterate quickly.

Many businesses start by targeting a particular group of internal developers—often mobile. This approach allows for some initial mistakes, learning and corrections as the team becomes more knowledgeable about APIs. We recommend a “fail-fast” approach. Failing is not a terrible thing; taking a long time to recognize it is. Starting internally also promotes a lighter governance model. A second internal audience or other lines of business may follow the initial stage to obtain further experience.

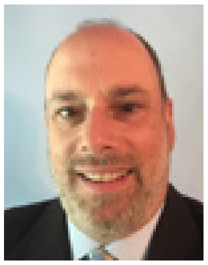
The next stage is to expand to partners. Typically, companies start with known partners who they want to engage in a new type of interaction that can be facilitated through APIs. This stage introduces additional governance and requires tightening up security, privacy and scaling. Plan well for change and versioning of your APIs. A second phase of this stage is to start creating APIs to enable new partner onboarding.

After that, the next stage is to go public. Initially, this stage will involve only APIs that enable already publicly available information—probably similar information to the information available on the current website. As time progresses, new and innovative cross-enterprise solutions will evolve, driving additional revenue and incenting further exploration of the API channel.

As we move into the API economy, there are huge opportunities for new and innovative solutions. The companies that derive the most value from those opportunities will have their business and IT organizations closely aligned, working together to drive success. IBM would like to be your partner on this journey, sharing our expertise and experiences to help maximize the value of APIs for your enterprise.

To understand more about the IBM perspective on the API Economy, visit the IBM API economy and Digital Transformation websites. IBM API Connect is a complete foundation to create, run, manage and secure APIs. You can find more information about IBM API Connect at the API Connect website and download a trial version here.

About the author



Alan Glickenhouse,
API Business Strategist

Alan Glickenhouse is a business strategist on the IBM API Connect offering management team. He joined IBM in 1981 and has held numerous positions in sales, technical sales, marketing, development and technical support. On the API Connect team, Alan assists clients in all industries with their business strategy for APIs, understanding their business direction and existing environment (both business and technical), and helps businesses successfully adopt an API strategy that fits their environment. Alan has an A.B. from Vassar College in Computer Mathematics and has several SOA certifications. Contact him at glick@us.ibm.com or follow @ARGlick.

[1] “GoDaddy Joins Newly Unified Node.js Foundation,” June 18, 2015, https://aboutus.godaddy.net/newsroom/news-releases/news-releases-details/2015/GoDaddy-Joins-Newly-Unified-Nodejs-Foundation/default.aspx

[2] Ó Maidin, Cian, “Why Node.js is becoming the go-to technology in the Enterprise,” NodeCrunch blog, March 10, 2014, www.nearform.com/nodecrunch/node-js-becoming-go-technology-enterprise

[3] Paul, Ryan, “A behind-the-scenes look at LinkedIn’s mobile engineering,” Oct. 2, 2012, http://arstechnica.com/information-technology/2012/10/a-behind-the-scenes-look-at-linkedins-mobile-engineering/2